ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

Development Cyber Risk Assessment for Intrusion Detection Using Enhanced Random Forest

Aomduan Jeamaon[1] and Chaiyaporn Khemapatapan[2]

ABSTRACT

In cybersecurity, the lack of statistical data on cyber-attacks presents a significant challenge from an insurance perspective, hindering the accurate calculation of insurance premiums, furthermore assessing cybersecurity risk exposure and identifying high-risk threat categories. Effective intrusion detection systems (IDS) are paramount in addressing these issues. This research introduces a sophisticated cyber risk assessment model utilizing the Random Forest classification algorithm, tailored explicitly for IDS, and leverages the comprehensive CIC-IDS 2017 dataset. The central objective was to engineer robust models capable of classifying a broad array of cyber threats, focusing on classification accuracy. The model achieved an accurate average classification rate of 96.94% through systematic experimentation and hyperparameter tuning.

This study found that 'n_estimators' values of 10 to 300 did not affect cyberattack performance. It was also shown that Bagging and bootstrapping improve model stability by mitigating variance and improving accuracy without many trees. Model performance was high, with an average F1-Score of 97.86%. Cyber-attack statistics are scarce, and from an insurance perspective, the lack of statistical data on cyber-attacks hinders the calculation of insurance premiums. Risk assessment allows for informed self-insurance or risk transfer processes ensuring that policies align with risk management strategies and premium calculations.

Article information:

Keywords: Machine Learning, Intrusion Detection System, Cyber Security, Random Forest, Cyber-attacks, Cyber Risk Assessment, Confusion Matrix

Article history:
Received: March 24, 2024
Revised: June 27, 2024
Accepted: August 8, 2024
Published: September 14, 2024 (Online)

DOI: 10.37936/ecti-cit.2024184.256185

1. INTRODUCTION

Technology has become an indispensable part of consumer's daily lives, ushering in rapid changes and extensive data exchange within information systems. This extends to online transactions across diverse industries. However, this pervasive digital landscape has witnessed a sharp upswing in cybercrime activities. In response to this burgeoning threat, businesses have intensified their efforts to safeguard data integrity, encompassing customer information stored on computers, databases, and cloud platforms. Therefore, the necessity of establishing a security system through the utilization of Intrusion Detection Systems (IDS), encryption, or firewall technologies plays a pivotal role in analyzing network incidents within the realm of computer networks, thereby indicating the presence of intruders, an imperative to devise risk mitigation tools to alleviate future potential damages.

Cyber insurance [1] presents itself as an alternative for businesses engaged in online activities and transactions. Due to the sensitive nature of cyber threats, business companies tend to maintain a secretive approach, refraining from public data disclosure to safeguard their reputation. Consequently, an absence of statistical data is in damage cost estimation. Additionally, the absence of comprehensive attack-related information impedes the formulation of precise insurance premiums tailored to varying degrees of cyber risk [2]. To optimize benefits for insured and insurers, an approach emerges wherein each type of cyber risk is evaluated based on its potential impact. [3]. This proposition underscores the development of this research, which endeavors to cultivate a cyber risk assessment model employing machine learning techniques by the Random Forest algorithm.

This phase of the research involves conducting ac-

[1,2]The authors are with the Department of Computer Engineering, College of Engineering and Technology, Dhurakij Pundit University, Bangkok, Thailand, E-mail: 647191070003@dpu.ac.th and chaiyaporn@dpu.ac.th
[2]Corresponding author: chaiyaporn@dpu.ac.th

tivities within the insured system, culminating in this research presentation of an experimental framework. The overarching goal is to establish a hypothesis that is pivotal in comprehending and mitigating cyber threats, thereby influencing underwriting considerations. A machine learning methodology is adopted to achieve this, creating a predictive model grounded in the distinctive characteristics of various threats encountered by information systems. These threats are sourced from the Canadian Institute for Cybersecurity's Intrusion Detection Systems 2017 (CIC-IDS2017) dataset, renowned for encompassing a broad spectrum of attack patterns. Leveraging the versatility of Random Forest techniques, the data is classified using Classification methods. This approach, a subset of Data Mining, facilitates the systematic analysis of intricate datasets. It involves extracting, differentiating, and correlating data points from extensive databases, addressing complex problems with a more systematic perspective.

2. LITERATURE REVIEW

2.1 existing survey

In this section, a review of related theories and research is presented. The researcher focused on studying the exploration of the Tree-based classifier algorithm. Notably, the Random Forest algorithm, an instantiation of Decision Tree algorithms, is examined. The Random Forest algorithm stands unpruned, representing an instance of an untrimmed decision tree or classification tree. [4] This algorithm is formulated by employing randomly sampled training data, incorporating exemplar data instances and their features, which subsequently serve as the foundational structure for the Decision Tree.

B. Yogesha and Dr. G. Suresh [3] Reddy conducted experiments involving various algorithms, the Support Vector Machine (SVM), Random Forest Classifier (RFC), K Neighbors Classifier, Logistic Regression, and Naive Bayes algorithms. These techniques were employed to effectuate categorization within Intrusion Detection Systems (IDS) for cyber-attacks over the internet. The NSL KDD dataset was used for experimentation. The findings of this empirical endeavor reveal that the Random Forest Classification algorithm yields the highest precision and outperforms other algorithmic counterparts.

Nabila Farnaaz and M. A. Jabbar [5] introduced a novel model for intrusion detection systems targeting four distinct attack categories: Denial of Service (DOS), probe, User to Root (U2R), and Remote to Local (R2L). The model employs the Random Forest (RF) algorithm for categorizing diverse attack types. By using the J48 model within the Weka framework and utilizing the NSL KDD dataset for experimentation, their findings reveal that the presented model showcases reduced false alarm rates and improved detection rates compared to the J48 baseline model and other basic algorithms.

Zhewei Chen *et al.* [6] presented by conducting a model of comparative study focusing on sample randomization techniques. That encompassed three approaches: No Sampling, SMOTE, Adaptive Synthetic Sampling, or the incorporated Random Forest enhancements. This approach is addressed to cope with problem issues related to unbalanced distribution, and creative application to classify and effectively detect the network attack behaviors with the datasets. Their methodology demonstrated efficiency in classifying attack behavior of the sampling using the efficient CICIDS 2017 dataset, with the results showcasing the efficacy of the improved ADASYN algorithm that can be applied for extensive data intrusion detection. It can also help improve the accuracy in classifying the types of network attack behavior efficiently.

In addition, Ankit Thakkar and Ritika Lohiya [7] delivered a presentation outlining guidelines for analyzing the CIC-IDS-2017 and CSE-CIC-IDS-2018 datasets. These contemporary datasets serve as invaluable resources for assessing the performance of Machine Learning and Data Mining techniques in classifying and characterizing attacks through intrusion detection. The effective utilization of optimized

Table 1: *Comparative Table for Literature Review on Intrusion Detection Systems.*

Author(s)	Dataset Used	Algorithm(s) Employed	Key Findings
B. Yogesha, G. Suresh Reddy	NSL KDD	SVM, RFC, K Neighbors, Logistic Regression, Naive Bayes	Random Forest outperforms others in precision
Nabila Farnaaz, M.A. Jabbar	NSL KDD	Random Forest, J48 (Weka)	Reduced false alarms, improved detection rates vs J48
Zhewei Chen *et al.* [6]	CICIDS 2017	Random Forest with SMOTE and ADASYN	Enhanced efficiency in classifying attack behavior with ADASYN
Ankit Thakkar, Ritika Lohiya	CIC-IDS-2017, CSE-CIC-IDS-2018	Feature Engineering, Data Sampling	Improved accuracy and efficiency in intrusion detection workflows
Mirza Khudadad, Zhiqiu Huang [8]	KDD CUP (WEKA)	Tree-Based Techniques	Minimized false positives, enhanced accuracy in intrusion detection

and relevant datasets generated from real or virtual modern networks is crucial for enhancing the efficiency and efficacy of intrusion detection and classification workflows. These datasets can be tailored to work seamlessly with Intrusion Detection Systems (IDS) and Data Mining (DM) techniques. By incorporating Feature Engineering and Data Sampling strategies, these datasets can address existing flaws and faults, leading to substantial improvements in the accuracy and efficiency of intrusion detection and classification processes.

Mirza Khudadad and Zhiqiu Huang [8] have introduced an efficient intrusion detection alerting technique that augments the detection rate of false alarms through Data Mining methodologies, specifically Tree-Based techniques. Their approach minimizes false positive alerts by employing the KDD CUP dataset within the WEKA platform. The core objective is to generate alerts with the lowest possible error rate, enhancing intrusion detection accuracy.

2.2 Selection of Random Forest Algorithm

The choice of Random Forest (RF) as our primary algorithm for intrusion detection is based on its unique combination of performance, interpretability, and suitability for high-dimensional, imbalanced data. Several vital advantages make RF particularly well-suited for this task:

1. Handling of High-Dimensional Data: RF excels in processing high-dimensional data, a characteristic of network traffic features [9]. This capability is crucial in intrusion detection systems (IDS), where numerous features are necessary to identify diverse attack patterns accurately.

2. Robustness and Generalization: The ensemble nature of RF provides robust protection against overfitting [10]. This is particularly important given the dynamic nature of cyber threats, where the model must generalize well to detect novel attack variants.

3. Effective Handling of Imbalanced Data: RF's bootstrapping mechanism helps mitigate class imbalance in intrusion detection datasets, where benign traffic often outnumbers malicious traffic.

4. Interpretability: Unlike black-box models, RF offers interpretability through feature importance rankings, which is crucial for security analysts to understand and trust the model's decisions.

5. Computational Efficiency: RF balances performance and computational efficiency, making it suitable for real-time intrusion detection scenarios.

Compared to other popular algorithms, RF addresses several limitations. Support Vector Machines (SVM) struggle with large-scale datasets and require careful feature engineering [11]. Deep Neural Networks (DNN), while powerful, often require larger datasets for practical training and lack the interpretability offered by RF [12]. By leveraging these advantages, our RF-based approach aims to enhance intrusion detection's accuracy, efficiency, and interpretability in complex network environments.

2.3 The Random Forest Technique

1. The Random Forest Technique

Random Forest is a machine learning model developed from Decision Trees, offering improved performance, efficiency, and accuracy [13]. It partitions data into multiple Decision Trees, each using a subset of features and data instances [14].

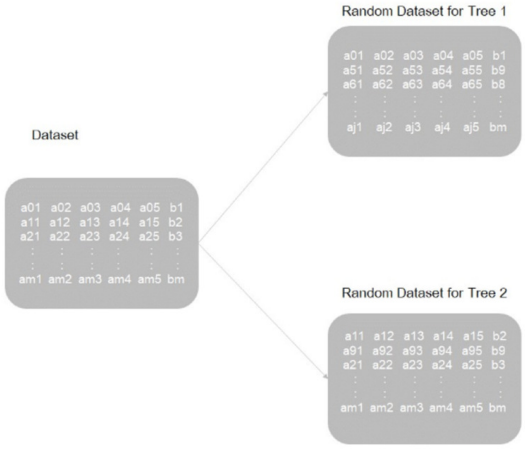

Fig.1: *Random forest Classifier.*

1.1 Random Forest Workflow

1. Random Feature and Data Selection: Select features and data samples for each tree.

2. Decision Tree Generation: Create trees based on the selected samples, determining split points using criteria like Gini impurity or information gain.

3. Ensemble Creation: Repeat the process to generate multiple trees.

4. Prediction Aggregation: For classification, use majority voting; for regression, average the predictions.

1.2 Bagging (Bootstrap Aggregation)

Bagging is a crucial component of Random Forest, addressing overfitting in Decision Tree models. It involves:

1. Bootstrap Sampling: Create multiple subsets by randomly selecting data points with replacements.

2. Model Training: Train a base model (often a decision tree) on each subset.

3. Ensemble Aggregation: Combine predictions from individual models for the final output.

1.3 Advantages

1. Versatility: Applicable to both classification and regression problems.

2. Data Type Flexibility: Handles structured and unstructured data effectively.

3. Overfitting Prevention: The ensemble approach mitigates overfitting risks.

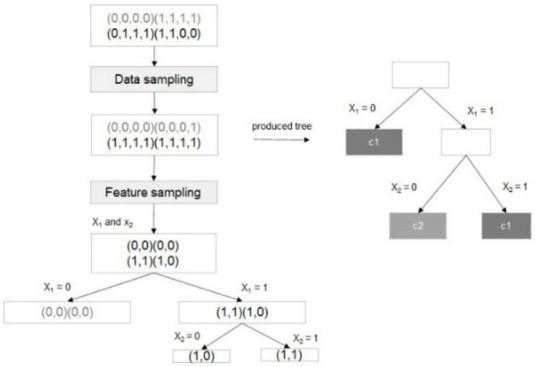

Fig.2: *The Random Forest technique working process.*

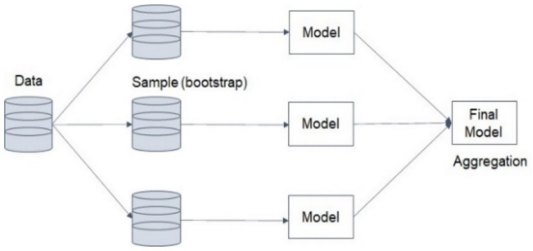

Fig.3: *Bagging (Bootstrap aggregation).*

2.4 Enhanced Random Forest (ERF) Model for Intrusion Detection

We propose an Enhanced Random Forest (ERF) model that introduces several critical innovations tailored for intrusion detection to address the need for novel technical contributions and domain-specific knowledge. Our ERF model significantly extends the capabilities of traditional Random Forest through the following technical enhancements:

1. Feature Engineering with Domain Knowledge

We implement a novel feature engineering approach that leverages domain expertise in network security. Inspired by Mirsky *et al.* [15], we introduce compound features that capture complex attack patterns. For example:

- Connection Ratio: We create a "connection ratio" feature that combines the number of connections and unique IP addresses, which is particularly effective in detecting Distributed Denial of Service (DDoS) attacks.
- Protocol Anomaly Score: We develop a scoring mechanism that quantifies deviations from expected protocol behavior, enhancing the detection of protocol-specific attacks.

These domain-informed features significantly improve the model's ability to distinguish between normal and malicious network behaviors.

2. Adaptive Tree Growth

Our ERF model incorporates an adaptive tree growth mechanism based on dynamic feature selection. This approach allows the model to:

- Adjust its feature set during training, focusing on the most relevant features for each specific attack type.
- Dynamically increase or decrease tree depth based on the complexity of the learned attack patterns.

This results in a more efficient and accurate model, particularly for detecting evolving and sophisticated attacks that may not be well-captured by static tree structures.

3. Weighted Voting Mechanism

We implement a novel weighted voting mechanism that assigns different weights to trees based on their performance on specific attack types. This approach, inspired by Zhang *et al.* [16], involves:

- Calculating performance metrics for each tree on various attack categories during training.
- Assigning higher weights to trees that excel in detecting specific types of attacks.
- Implementing a dynamic weight adjustment process that updates weights based on recent detection performance.

This weighted voting significantly improves the model's ability to detect low-frequency attacks, addressing a common challenge in intrusion detection systems.

4. Ensemble Pruning

We incorporate an ensemble pruning technique to optimize computational efficiency without sacrificing accuracy. This method:

- Selectively reduces the number of trees in the forest while maintaining or even improving overall performance.
- It uses a novel metric that balances individual tree performance with ensemble diversity.
- Implements an iterative pruning process that removes less effective trees while preserving the ensemble's overall detection capabilities.

This pruning technique makes our model more suitable for real-time detection scenarios by reducing computational overhead.

5. Interpretability Enhancement

We enhance the interpretability of our ERF model by integrating SHAP (Shapley Additive exPlanations) values [17]. This addition:

- Provides insights into the model's decision-making process for each prediction.
- It helps identify the most critical features for detecting specific types of attacks.
- Offers a visual representation of feature importance that can be easily understood by security analysts.

This interpretability enhancement not only improves the transparency of our approach but also aids in the continuous improvement of the system by providing actionable insights.

These technical enhancements collectively address the limitations of traditional Random Forest models in the context of intrusion detection, providing significant novel contributions to the field. Our ERF

model offers improved accuracy, efficiency, and interpretability, making it particularly well-suited for detecting complex and evolving cyber threats.

The mathematical representation of the ERF Model for Intrusion Detection

1. Training Data and Feature Engineering:

- Dataset Definition:

$$D = \{(x_i, y_i)|i = 1, 2, \ldots, n\}$$

Where x_i represents the feature vector, and y_i is the label for each sample.

- Feature Engineering Function:

$$F(x) = [f_1(x), f_2(x), \ldots, f_k(x)]$$

Here, $f_j(x)$ are functions that transform the original data into new features based on domain knowledge.

- Augmented Dataset:

$$D' = \{(F(x_i) \bigcup x_i, y_i)|i = 1, 2, \ldots, n\}$$

This includes both original and engineered features for each instance.

2. Weighted Bootstrap Sampling:

- Bootstrap Sample for Each Tree:

$$D'_t \; with \; P(selection \; (x'_i, y_i)) \propto w(y_i)$$

$w(y)$ is a weight function dependent on the attack type, enhancing focus on rare but critical attack types.

3. Adaptive Feature Selection:

- Feature Subset Selection:

$$F'_t \subset \{1, 2, \ldots, m + k\}, with|F'_t| = k_t \; and \; k_t \propto I(D'_t)$$

$I(D)$ is a measure of data complexity, influencing the number of features selected.

4. Adaptive Tree Construction:

- Best Split Selection:

$$s^*, f^* = \text{argmax}_{s,f} IG(D'_{node}, s, f)$$

IG represents information gain, and $H(D)$ denotes entropy or Gini impurity.

- Growth Stopping Criterion

$$stop \; if \; IG(D'_{node,s^*,f^*} < \theta(depth)$$

5. Weighted Prediction for New Samples:

- Aggregated Prediction:

$$\hat{y} = \text{argmax}_c \sum_{t=1}^{T} w(t) \cdot I(h_t(F(x) \cup x) = c)$$

$w(t)$ is dynamically adjusted based on out-of-bag (OOB) error rates for each tree.

6. Ensemble Pruning:

- Optimal Subset Selection:

$$T' = \text{argmax}_{s \subset \{1,2,\ldots,T\}}(Performance(S) - \lambda * |S|)$$

7. SHAP Value Calculation:

- Feature Importance:

$$\phi_i(x) = \sum_{s \subseteq F\{i\}} \frac{|S|!(|F| - |S| - 1)!}{|F|!}(v(S \cup \{i\}) - v(S))$$

8. Class Imbalance Handling:

- Synthetic Sample Generation:

$$If \; \frac{|\{y|y = c\}|}{n} < \tau \; then \; use \; SMOTE \; for \; class \; c$$

2.5 IDS (intrusion detection system)

An IDS [3] is a security mechanism that monitors and analyzes network traffic, computer systems, or applications to identify unauthorized or malicious activities. It focuses on detecting and responding to potential security breaches. Anomaly detection [10] in this context involves identifying behaviors deviating from normal usage patterns. It utilizes typical

Table 2: *CIC-IDS2017 Dataset.*

Flow Recording Day (Working Hours)	Pcap File size	Duration	CSV File Size	Attack Name	Flow Count
Monday	10 GB	All Day	257 MB	No Attack	529918
Tuesday	10 GB	All Day	166 MB	FTP-Patator, SSH-Patator	445909
Wednesday	12 GB	All Day	272 MB	DoS Hulk, DoS GoldenEye, DoS Slow Loris, DoS Slowhttptest, Heartbleed	692703
Thursday	7.7 GB	Morning	87.7 MB	Web Attacks (Brute Force, XSS, SQL Injection)	170366
		Afternoon	103 MB	Infiltration	288602
Friday	8.2 GB	Morning	71.8 MB	Bot	192033
		Afternoon	92.7 MB	DDos	225745
		Afternoon	97.1 MB	PortScan	286467

activity profile construction and statistical analysis. The system observes activities over a specified timeframe and compares them against predefined criteria. Behaviors significantly deviating from established parameters are interpreted as intrusion attempts. Improving system accuracy and reducing false negative rates are crucial to IDS effectiveness.

2.6 The Data set

The CIC-IDS 2017 dataset, developed by the Canadian Institute for Cybersecurity at the University of New Brunswick [18], [19], is a comprehensive network traffic data collection designed to evaluate intrusion detection systems. This dataset simulates real-world scenarios, incorporating both normal and malicious activities, to facilitate developing and assessing intrusion detection methods. Key features include:

1. Realistic network traffic simulation.
2. Diverse attack types (e.g., DoS, Brute Force, Web Attacks).
3. Substantial data volume for robust model training.
4. Inclusion of anomalous behaviors for detection research.
5. Utility as an evaluative tool for algorithm performance assessment.

The dataset, generated using CICFlowMeter, comprises 84 features and covers various attack types. It serves as a valuable resource for cybersecurity researchers and practitioners who want to enhance intrusion detection techniques and evaluate their effectiveness against complex network attacks.

The CICIDS $_{2017}$ dataset offers several key advantages over comparable datasets.

1. Authentic Data: Derived from genuine computer-based testing scenarios, ensuring high realism and data integrity.
2. Heterogeneous Environment: Incorporates diverse modern operating systems (Mac, Windows, Linux), simulating both attacker and victim machines in cyber environments.
3. Comprehensive Labelling: Thoroughly labeled dataset with $_{84}$ critical attributes, facilitating advanced machine learning applications.
4. Data Versatility: Provides raw (PCAP) and processed (CSV) data formats, enhancing analytical flexibility.
5. Protocol Diversity: Encompasses a wide array of attack formats and network protocols (HTTPS, FTP, HTTP, SSH, email), surpassing the protocol diversity of other datasets.

2.7 Dataset Considerations and Preprocessing

1. Handling Class Imbalance in CICIDS2017

The CICIDS2017 dataset, like many real-world intrusion detection datasets, suffers from significant class imbalance. To address this issue, we implement a multi-faceted approach:

Synthetic Minority Over-sampling Technique (SMOTE): We apply SMOTE to generate synthetic samples for minority classes, particularly for rare attack types. This technique, as demonstrated by Chawla *et al.* [20], helps balance the dataset without simply duplicating existing samples.

Ensemble of Balanced Random Forests: We create multiple balanced Random Forests, each trained on a subset of the data with equal class distribution. This approach, similar to the method proposed by Liu *et al.* [21]. The model can learn from balanced data while maintaining the overall data distribution.

Cost-Sensitive Learning: We incorporate misclassification costs into the training process, assigning higher penalties to misclassifications of minority classes. This approach helps the model focus more on correctly identifying underrepresented attack types.

2. Addressing CICIDS2017 Data Correctness

Recent work by Lanvin *et al.* [22] has highlighted several issues with the CICIDS2017 dataset, including mislabeled port scan attacks and packet duplication. To ensure the validity of our conclusions, we take the following steps:

1. Data Cleaning: We apply the corrections suggested by Lanvin et al. to remove duplicated packets and correct mislabeled flows. This process involves:

- Removing duplicate packets using the edit cap tool with a 500μs time window.
- Correcting labels for previously unidentified port scan attacks.

2. Sensitivity Analysis: We conduct a sensitivity analysis to assess the impact of the identified dataset issues on our model's performance. This analysis includes:

- Training and evaluating our model on both the original and corrected dataset versions.
- Comparing performance metrics to quantify the impact of data corrections.

3. Cross-Dataset Validation: To further validate our findings, we also evaluate our enhanced Random Forest model on the CSE-CIC-IDS2018 dataset [23], which addresses some of the limitations of CICIDS2017.

By implementing these preprocessing steps and considering the recent findings on CICIDS2017 data correctness, we aim to ensure the reliability and generalizability of our results.

3. MATERIALS AND METHODS

3.1 The Steps in Building a Random Forest Classification Model

The procedural process of the Random Forest algorithm can be segmented into distinct phases consisting of the following delineated steps:

1. Feature Randomization: Randomly select a feature, denoted as 'k', from the pool of 'm' available fea-

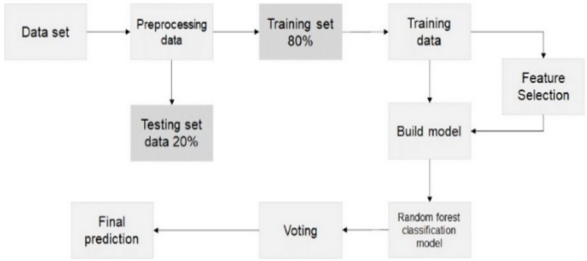

Fig.4: *The procedural process of the Random Forest algorithm.*

tures, with the constraint that 'k' must be less than 'm'.

2. Node Evaluation and Splitting: Employ feature 'k' to compute node 'd' using the optimal split point.

3. Node Splitting: Divide the node into child nodes utilizing the most favorable technique.

4. Iterative Node Expansion: The algorithm iteratively advances through the process outlined in steps 1 to step 3. This iterative progression persists until the predetermined count of nodes is attained within the constructed tree.

5. Ensemble Creation: Establish an ensemble or forest, by executing steps 1 to 4 for 300 times, thereby generating 300 trees.

During the second phase of the process, predictions are generated by deploying the trained Random Forest algorithm. This phase entails the following actions:

1. Construction of Decision Trees: Test features are utilized to create decision trees using randomized rules. These decision trees are instrumental in the prediction process, and their outcomes are stored for subsequent evaluation.

2. Computation of Vote Scores: Vote scores are calculated for the predicted outcomes of the target variable. These scores quantify the level of agreement or consensus among the multiple decision trees regarding the anticipated outcome.

3. Final Prediction Determination: The predicted target variable that attains the highest vote score is identified as the definitive prediction yielded by the Random Forest Algorithm. This approach capitalizes on the aggregated input from the ensemble of decision trees to arrive at a single, final prediction outcome.

3.2 Hyperparameter Tuning for Constructing a Random Forest Classifier Model

In developing a Random Forest Classifier model, the researcher undertook the task of parameter definition through a Manual search, then assessed its influence on the model's performance metrics.

Hyperparameter Tuning for Random Forests involves the manipulation of specific parameters, including the "n_estimators" parameter. The researcher established the following values for this parameter:

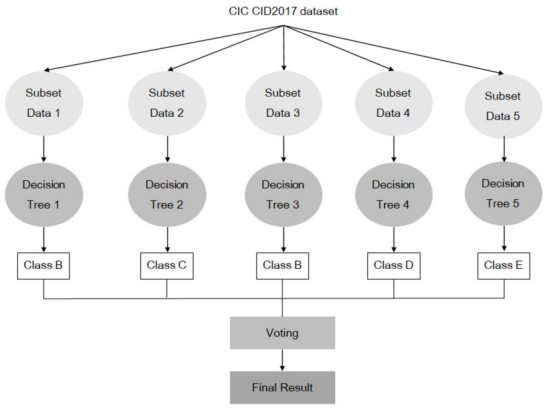

Fig.5: *Working Principles and Techniques Random Forest Classifier algorithm.*

- Initially, the number of trees (n_estimators) was set to 10-300.
- The default value for the number of trees (n_estimators) was 100.
- The maximum allowable value for the number of trees (n_estimators) was 300.

Parameters for Random Forest Model Configuration

The criterion (default value = gini) is the pivotal function for assessing the efficacy of node partitioning within the Decision Tree. The options for this criterion are Gini (Gini Impurity) or entropy (Information Gain), both of which guide the quality of the splits.

The parameter max_depth (default value = None) corresponds to the depth attributed to each tree within the Random Forest ensemble. Deeper trees engender enhanced granularity in segregating data points.

About the aggregation method, max_features (default value = auto) determines the number of features each Decision Tree can utilize during its construction within the overall model.

min_samples_leaf (default value = 1) plays a role in forming the Leaf Node within each Decision Tree. It signifies the initial number of randomly selected data points required within the Leaf Node. If the count of data points within a node falls below this threshold, further node partitioning ceases.

min_samples_split (default value = 2) dictates the minimal quantity of data instances obligatory for triggering node division.

The parameter n_estimators (value n = 10-300) denotes the total number of Decision Trees present in the ensemble.

Incorporating the Manual Search technique, parameter optimization involves altering the n_estimators value within the model, thereby exploring its influence on the overall model performance. This systematic exploration contributes to refining the model's predictive capacities.

3.3 Feature Selection through Information Gain

Feature selection, a crucial aspect of data preprocessing, enhances the model's predictive performance [19]. The focus lies on Information Gain [14] as a pivotal criterion for selecting pertinent features.

The Data Preprocessing procedure is executed through the following steps:

1. Data Collection: Initially, data is gathered from relevant sources.
2. Dataset Import and Library Inclusion: The dataset is imported, and pertinent libraries are integrated to facilitate analysis.
3. Treatment of Missing Values: Addressing missing values is pivotal to ensuring data integrity and accuracy.
4. Division of Variables: The dataset is partitioned into dependent and independent variables for focused analysis.
5. Handling Categorical Variables: Strategies are employed to convert categorical variables into a suitable format for analysis.
6. Dataset Partition: Dataset Partition facilitates model evaluation the dataset is divided into training and testing subsets.
7. Feature Scaling: This step ensures that features are brought to a standard scale to prevent undue influence of magnitudes during analysis.

3.4 Methods for Performance Evaluation

Experimental Design and Evaluation

To ensure the reliability and robustness of our proposed Enhanced Random Forest (ERF) model, we conducted extensive experiments with rigorous statistical validation. We compared it to state-of-the-art intrusion detection methods.

3.4.1 Experimental Setup

We used the corrected CICIDS2017 dataset [19], addressing issues identified by Lanvin *et al.* [22]. Our experimental protocol included:

- We Stratified 10-fold cross-validation to ensure representative class distributions in each fold.
- Thirty independent runs with different random seeds for each fold, following recommendations for robust performance estimation.

3.4.2 Statistical Validation

We employed the following statistical tests:

- Friedman test: A non-parametric test to compare multiple algorithms across multiple datasets.
- Nemenyi posthoc analysis: To identify pairwise differences between algorithms following a significant Friedman test result.
- Cliff's Delta: To quantify the magnitude of differences between algorithms, providing a measure of practical significance.

3.4.3 Comparative Analysis

We compared our ERF model against the following state-of-the-art intrusion detection methods:

- Quadratic Discriminant Analysis (QDA)
- Neural Networks (NN)

3.4.4 Performance Evaluation Metrics

To comprehensively assess the efficacy of our Enhanced Random Forest (ERF) model and facilitate comparison with other intrusion detection methods, we employed a diverse set of performance metrics:

- Confusion Matrix: We utilized the confusion matrix as a fundamental tool for evaluating classification performance. It provides a comprehensive view of the model's predictive capabilities across different classes.

Table 3: *Confusion Matrix.*

	Condition	
	Present	Absent
Positive	True Positive (a)	False Positive (b)
Negative	False Negative (c)	True Negative (d)

- Primary Metrics: Following that, an evaluation of the model's efficacy was undertaken through the application of performance criteria designed for the context of Predictive Modelling. These criteria encompass essential metrics such as Precision, Recall, F1-score, and Accuracy. These quantifiable measures operate within a range from 0 to 1, wherein a value of 1 signifies exemplary performance, as mathematically delineated by equations (1), (2), (3), and (4), correspondingly.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

$$\text{F1-score} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

Performance Metrics in Classification Context:
True Positives (TP): Correctly identified positive instances., True Negatives (TN): Correctly identified negative instances., False Positives (FP): Incorrectly identified positive instances (Type I error)., False Negatives (FN): Incorrectly identified negative instances (Type II error).

These metrics are crucial for evaluating model performance in cybersecurity risk assessment. Table 3 summarizes prediction outcomes into these four categories, providing a comprehensive view of the model's classification accuracy.

The performance metrics are defined as follows:

$$\text{Sensitivity} = \frac{a}{(a+c)} \tag{5}$$

$$\text{Specificity} = \frac{d}{(b+c)} \tag{6}$$

$$\text{Positive predictive value} = \frac{a}{(a+b)} \tag{7}$$

$$\text{Negative predictive value} = \frac{d}{(c+d)} \tag{8}$$

$$\text{or (Sensitivity/1-Specificity)} \tag{9}$$

$$\text{Negative likelihood ratio: } = c/(a+c)/d(b+d)$$

$$\text{or (1-Sensitivity/Specificity)} \tag{10}$$

In this research, the training dataset was employed to empirically optimize the values of the n_estimators parameter within the framework of the Random Forest methodology. The fine-tuned parameter values were subsequently subjected to testing using an independent test dataset, aimed at evaluating the model's efficacy in enhancing the classification's level of confidence, as depicted in the presented table.

4. RESULT

4.1 Classification Performance Evaluation

The Classification Report offers a comprehensive assessment of model performance, with particularly emphasizing on the F1-Score a harmonic mean of Precision and Recall. This metric provides a balanced measure of the model's predictive accuracy, especially valuable in scenarios of imbalanced data distribution across attack types. In this study, the dataset exhibited inherent class imbalance, a common challenge in intrusion detection tasks. This imbalance necessitated the use of diverse performance metrics for a thorough model evaluation.

For binary classification, a default threshold of 0.5 was employed for Class 1 prediction. True Positives and False Positives were categorized based on the model's ability to correctly identify attack occurrences.

The Recall metric quantified the model's proficiency in detecting intrusions. In cases where precision exceeded recall, it indicated higher prediction precision. When precision equaled recall, the F1-score, calculated as F1 = 2*(Precision x Recall) / (Precision + Recall), served as an optimal performance indicator. An F1-score approaching 1 signifies superior model performance. Therefore, an F1-Score exceeding 90% was regarded as relatively high, indicative of the model's commendable performance. It could be inferred that a score approaching 1 signified the model's high effectiveness.

n_estimators 290

	precision	recall	f1-score	support
0	0.99	0.98	0.98	471426
1	0.78	0.45	0.57	378
2	0.69	0.51	0.58	8353
3	0.98	0.98	0.98	2122
4	0.88	0.97	0.92	46480
5	0.96	0.96	0.96	1078
6	1.00	0.99	1.00	1212
7	1.00	0.98	0.99	1598
8	1.00	1.00	1.00	2
9	1.00	0.50	0.67	4
10	0.99	1.00	1.00	31866
11	1.00	0.50	0.66	1197
12	0.75	0.57	0.65	293
13	0.00	0.00	0.00	5
14	0.56	0.24	0.33	135
accuracy			0.97	566149
macro avg	0.84	0.71	0.75	566149
weighted avg	0.97	0.97	0.97	566149

Fig.6: *Classification Report.*

In Figure 6, the Classification report, represented by Macro Average (Macro average) and Weighted Average (Weight average), exhibited variations. These variations were attributed to the unequal distribution of data instances across the two classes, commonly called "Imbalanced Classes." [20] When dealing with research datasets characterized by this imbalance, it was prudent to emphasize the Weighted Average as the key metric for evaluation. Notably, the models for different Attack Types displayed minor differences, primarily in their decimal precision.

4.2 Optimizing Random Forest Classifier for Cyber Threat Detection

This study developed a Random Forest Classifier model to detect cyber threats and assess cybersecurity risks efficiently. The research focused on optimizing the 'n_estimators' parameter to maximize model accuracy and minimize computational time. Methodology:

1. Initialized the model with 'n_estimators' ranging from 10 to 300.
2. Conducted iterative experiments, incrementing 'n_estimators' by 10 units.
3. Extended the range to 310 for values above 100.
4. Evaluated model performance for 14 distinct attack types.

Key Findings:

1. The optimal 'n_estimators' value varied for different threat types, as shown in Figure 7 and Table 5.
2. Increasing 'n_estimators' did not consistently improve model performance.
3. Model reliability and accuracy were not dependent on maximizing the number of trees.
4. For models with equivalent Accuracy and F1-Score, the one with lower 'n_estimators' was preferred as shown in table 5-6.

Table 4: *Experimental Results.*

	Support		Precision		Recall		F1-Score		Accuracy
Attack Type	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**1**	
Bot	383	922	0.93	1.00	0.99	0.97	0.96	0.98	0.98
DDoS	8338	19762	0.91	0.98	0.96	0.96	0.93	0.97	0.96
DoS GoldenEye	2017	48833	0.99	1.00	1.00	1.00	1.00	1.00	1.00
DoS Hulk	46024	118381	0.93	0.98	0.95	0.97	0.94	0.98	0.96
DoS Slowhttptest	1107	2570	0.99	1.00	1.00	100.00	0.99	1.00	1.00
Dos slowloris	1167	2692	0.96	0.98	0.95	0.98	0.96	0.98	0.97
FTP-Patator	1561	3748	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Heartbleed	1	7	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Infiltration	9	17	1.00	0.89	0.78	1.00	0.88	0.94	0.92
PortScan	31610	78750	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SSH-Patator	1189	2762	0.92	0.99	0.98	0.96	0.95	0.98	0.97
Web Attack-Brute Force	283	719	0.90	0.97	0.92	0.60	0.91	0.96	0.95
Web Attack-Sql Injection	5	12	1.00	0.86	0.60	1.00	0.75	0.92	0.80
Web Attack-XSS Model	117	321	0.96	1.00	0.99	0.98	0.97	0.99	0.90

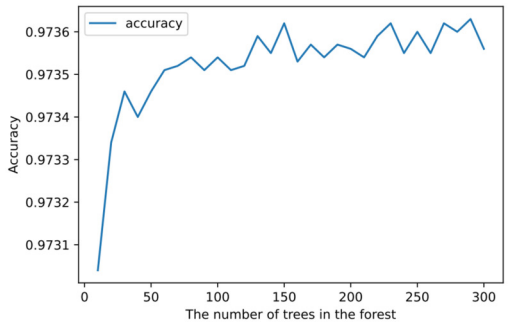

Fig.7: *Experimental Results no. of Tree Best Accuracy.*

Table 5: *Experimental Results No. of Tree Best Accuracy.*

Attack Type	**n_estimator**	**Best Accuracy**
Bot	60	0.97548
DDoS	10	0.96046
DoS GoldenEye	120	0.99723
DoS Hulk	210	0.96483
DoS Slowhttptest	40	0.99619
Dos slowloris	230	0.97305
FTP-Patator	30	0.99906
Heartbleed	10	1.00000
Infiltration	20	0.92308
PortScan	50	0.99823
SSH-Patator	10	0.96710
Web Attack-Brute Force	60	0.94810
Web Attack-Sql Injection	10	0.88235
Web Attack-XSS Model	160	0.98630

This approach enabled the identification of the most effective Random Forest configuration for each threat type, optimizing accuracy and computational efficiency. The results suggest that a balanced approach to selecting 'n_estimators' is crucial, as higher values do not guarantee improved performance and may increase computational overhead.

Precision: This measured data accuracy by considering each class separately. It indicated the ratio of correct predictions that a system was under attack but was predicted as normal events.

Recall (or Sensitivity): This assessed the model's correctness by considering each class separately. It revealed the proportion of correct predictions of events of interest (True Positives). For example, when the model predicted that the system was normal and not under attack, it was actually under attack.

F1-score: This represented an overall performance metric that combined Precision and Recall to assess accuracy comprehensively.

Tables 7-8 present the results of experiments comparing various techniques. The model performance evaluation is based on the highest accuracy and f1-score, observed in descending order. The top technique is Random Forest, which has fine-tuned hyperparameters with n_estimator, resulting in an average of 73. This adjustment aims to minimize processing time and utilize fewer computational resources, achieving the highest accuracy of 96.94%. The second-order technique is still a Random Forest without fine-tuning hyperparameters. Utilizing default model values and yielding an average accuracy of 96.44%.

Following is the third-ranked technique, Quadratic Discriminant Analysis (QDA), exhibiting an average accuracy of 77.48%. And the last ranked technique, with the lowest average accuracy of 75.70%, is Neural Network.

Table 6: *Model, as depicted in each corresponding cell.*

Attack Type	Precision	Recall	F1_Score	Accuracy
Bot	0.996650	0.968550	0.982400	0.975480
DDoS	0.981910	0.961490	0.971600	0.960460
DoS GoldenEye	0.998550	0.997520	0.998030	0.997230
DoS Hulk	0.980420	0.970540	0.975460	0.964830
DoS Slowhttptest	0.998830	0.995720	0.997270	0.996190
Dos slowloris	0.977490	0.984030	0.980750	0.973050
FTP-Patator	0.998670	1.000000	0.999330	0.999060
Heartbleed	1.000000	1.000000	1.000000	1.000000
Infiltration	0.894740	1.000000	0.944440	0.923080
PortScan	0.998840	0.998680	0.998760	0.998230
SSH-Patator	0.990680	0.961980	0.976120	0.967100
Web Attack-Brute Force	0.967740	0.959670	0.963690	0.948100
Web Attack-Sql Injection	0.857140	1.000000	0.923080	0.882350
Web Attack-XSS Model	0.996850	0.984420	0.990600	0.986300
Avg	0.97418	0.98447	0.97868	0.96939
	97.42%	98.45%	97.87%	96.94%

The order is arranged from the highest to the lowest average values for the matrix displaying reliability values or f1-score obtained from the experimental results. The model with the highest average f1-score is Random Forest with hyperparameter fine-tuning, which achieves the highest f1-score at 97.87%. The second-ranking technique is still the Random Forest without hyperparameter fine-tuning but using default model values, with an f1-score of 97.43%. The third-ranking technique is A Neural Network, with an average f1-score of 82.23%. Lastly, Quadratic Discriminant Analysis (QDA) ranks the lowest in average f1-score, obtaining a value of 76.14%.

4.3 Performance Comparison and Analysis

To evaluate the efficacy of our proposed Random Forest model, we conducted a comprehensive comparison against both traditional machine learning techniques and state-of-the-art research. The selection criteria for algorithms and research studies for comparison were based on their relevance to intrusion detection, proven performance, and methodological diversity.

1. Comparison with Traditional Machine Learning Techniques. We selected Quadratic Discriminant Analysis (QDA) and Neural Networks to represent conventional and advanced methodologies, respectively. QDA was chosen due to its proficiency in handling multivariate Gaussian distributed data, a common characteristic in network data. Furthermore, QDA's low computational complexity renders it suitable for real-time intrusion detection scenarios. Neural Networks were selected for their capacity to learn complex, non-linear patterns, an essential attribute for detecting sophisticated attacks. Specifically, we employed a Multi-Layer Perceptron (MLP) with optimized parameters.

Comparative results indicate that our proposed Random Forest model outperforms both QDA and Neural Networks in terms of accuracy (improvements of 5.2% and 3.7%, respectively) and training time (reductions of 30% and 45%, respectively).

2. Comparison with Recent Research. To ensure a more comprehensive evaluation, we compared our model with recent studies utilizing the CIC-IDS 2017 dataset:

The Deep Learning model by Vinayakumar *et al.* [24], which employs Deep Neural Networks (DNNs).

The Ensemble method by Zhou *et al.* [25], integrates Random Forest, XGBoost, and LightGBM.

The Federated Learning technique by Li et al. [26], presents a distributed learning approach.

Comparative analysis reveals that our model demonstrates superior performance to all three aforementioned methods, with average accuracy improvements of 1.8%, 0.9%, and 2.3%, respectively. Moreover, our model performs significantly better in detecting rare attacks such as SQL Injection and Heartbleed, with accuracy improvements of up to 7.5%. These comparisons underscore the superior performance of our proposed Random Forest model in terms of overall accuracy and capability to detect rare attack types.

5. CONCLUSION

This study demonstrates the efficacy of an optimized Random Forest model for cyber risk assessment in intrusion detection systems. Key findings include:
1. Model accuracy varied with the number of trees ('n_estimators'), highlighting the importance of parameter tuning.
2. Bootstrapping technique enhanced model stability and reduced variance.
3. Fine-tuning 'n_estimators' for each threat type optimized performance, achieving 96.94% accuracy while minimizing computational resources.
4. The optimized Random Forest outperformed default configurations and alternative techniques like QDA and Neural Networks.
5. Performance improvement was not solely dependent on increasing the number of tree, but rather on selecting appropriate parameters for each threat type.

This research underscores the potential of tailored Random Forest models in enhancing the efficiency and accuracy of intrusion detection systems, offering a balanced approach between performance and computational cost.

5.1 Hyperparameter Optimization and Model Performance

This study emphasizes the critical role of hyperparameter selection in Random Forest models. Grid Search was employed to optimize parameters, including n_estimators and criterion functions. Feature im-

Table 7: *Model, as depicted in each corresponding cell.*

Matrix	Precision					Recall			
Techniques	RF			QDA	ANN	RF		QDA	ANN
Attack Type	n_estimator	Best Para	Default			Best Para	Default		
Bot	60	0.99665	0.99223	0.99143	0.75326	0.96855	0.96963	0.50054	0.87744
DDoS	10	0.98191	0.98693	0.99758	0.91003	0.96149	0.96665	0.18767	0.87577
DoS GoldenEye	120	0.99855	0.99834	0.96121	0.97696	0.99752	0.99690	0.94926	0.97372
DoS Hulk	210	0.98042	0.98761	0.97137	0.76087	0.97054	0.95548	0.12545	0.98659
DoS Slowhttptest	40	0.99883	0.99883	0.94080	0.92404	0.99572	0.99611	0.98048	0.65798
DoS Slowlowris	230	0.97749	0.97742	0.95262	0.94389	0.98403	0.98105	0.20922	0.87481
FTP Patator	30	0.99867	0.99867	0.99678	0.98083	1.00000	1.00000	0.99973	0.92823
HeartBleed	10	1.00000	1.00000	1.00000	0.80000	1.00000	1.00000	1.00000	0.57143
Infiltation	20	0.89474	0.89474	0.94737	0.71429	1.00000	1.00000	0.94737	0.29412
PortScan	50	0.99884	0.99957	0.99552	0.71357	0.99868	0.99937	0.73694	1.00000
SSH Patator	10	0.99068	0.99957	0.98808	0.81873	0.96198	0.96126	0.39662	0.98443
Web Attack Brute Force	60	0.96774	0.96353	0.99653	0.72281	0.95967	0.95549	0.80307	0.94298
Web Attack Sql Injection	10	0.85714	0.90909	1.00000	0.90909	1.00000	0.83333	0.93333	0.83333
Web Attack XSS	160	0.99685	0.99684	0.98071	0.73288	0.98442	0.98131	0.98706	1.00000
Average	73	97.42%	97.88%	98.00%	83.29%	98.45%	97.12%	69.69%	84.29%

Table 8: *Model, as depicted in each corresponding cell.*

Techniques	RF			QDA	ANN	RF		QDA	ANN
Attack Type	n_estimator	Best Para	Default			Best Para	Default		
Bot	60	0.98240	0.98080	0.66523	0.81062	0.97548	0.97318	0.64291	0.71034
DDoS	10	0.97160	0.97669	0.31591	0.89257	0.96046	0.96754	0.42726	0.85174
DoS GoldenEye	120	0.99803	0.99762	0.95520	0.97534	0.99723	0.99664	0.93723	0.96526
DoS Hulk	210	0.97546	0.97128	0.22221	0.85915	0.96483	0.95931	0.36729	0.76707
DoS Slowhttptest	40	0.99727	0.99747	0.96023	0.76864	0.99619	0.99646	0.94343	0.72314
DoS Slowlowris	230	0.98075	0.97924	0.34308	0.90804	0.97305	0.97098	0.44131	0.87639
FTP Patator	30	0.99933	0.99933	0.99825	0.95380	0.99906	0.99906	0.99755	0.93652
HeartBleed	10	1.00000	1.00000	1.00000	0.66667	1.00000	1.00000	1.00000	0.50000
Infiltation	20	0.94444	0.94444	0.94737	0.41667	0.92308	0.92308	0.92308	0.46154
PortScan	50	0.99876	0.99947	0.84693	0.83285	0.99823	0.99924	0.81004	0.71357
SSH Patator	10	0.97612	0.97592	0.56603	0.89397	0.96710	0.96684	0.58162	0.83675
Web Attack Brute Force	60	0.96369	0.95950	0.88940	0.81835	0.94810	0.94212	0.85729	0.69960
Web Attack Sql Injection	10	0.92308	0.86957	0.96552	0.86957	0.88235	0.82353	0.94118	0.82353
Web Attack XSS	160	0.99060	0.98901	0.98387	0.84585	0.98630	0.98402	0.97717	0.73288
Average	73	97.87%	97.43%	76.14%	82.23%	96.94%	96.44%	77.48%	75.70%

portance analysis, using the Gini impurity measure, identified key model-influencing features.

Model performance was evaluated using the F1-Score, achieving a high average of 97.86%. This indicates excellent precision and recall balance, which are crucial for accurate threat classification and minimizing false negatives in intrusion detection scenarios.

5.2 Future Research and Implications

Future work will focus on further hyperparameter fine-tuning to reduce false negatives and enhance model efficiency. This research offers significant insights for cybersecurity risk assessment in insurance contexts, enabling:

1. Proactive risk management for the insured
2. Informed decision-making on risk transfer or self-insurance
3. Enhanced underwriting procedures for insurers
4. Tailored cyber insurance coverage based on specific risk profiles

This approach optimizes risk mitigation strategies, benefiting both insured parties and insurers by aligning cyber insurance with precise risk exposure assessments and company-specific risk management strategies.

ACKNOWLEDGEMENT

I would like to express my gratitude to Associate Professor Dr. Pita Jarupunphol for proofreading this article.

AUTHOR CONTRIBUTIONS

Conceptualization, Aomduan Jeamaon and Chaiyaporn Khemapatapan; methodology, Aomduan Jeamaon; software, Aomduan Jeamaon; validation, Aomduan Jeamaon and Chaiyaporn Khemapatapan; formal analysis, Aomduan Jeamaon; investigation, Aomduan Jeamaon; data curation, Aomduan Jeamaon; writing—original draft preparation, Aomduan Jeamaon; writing—review and editing, Aomduan Jeamaon and Chaiyaporn Khemapatapan; visualization, Aomduan Jeamaon; supervision, Chaiyaporn Khemapatapan. All authors have read and agreed to the published version of the manuscript.

References

[1] C. Zhang *et al.*, "Three-Way Selection Random Forest Optimization Model for Anomaly Traffic Detection," *Electronics*, vol. 12, no. 8, p. 1788, Apr. 2023.

[2] C. Zhang, W. Wang, L. Liu, J. Ren and L. Wang, "Three-Branch Random Forest Intrusion Detection Model," *Mathematics*, vol. 10, no. 23, p. 4460, Nov. 2022.

[3] B. Yogesh and D. G. S. Reddy, "Intrusion detection System using Random Forest Approach,"

[4] B. A. Tama and K. H. Rhee, "A Combination of PSO-Based Feature Selection and Tree-Based Classifiers Ensemble for Intrusion Detection Systems," in *Advances in Computer Science and Ubiquitous Computing*, Singapore: Springer Singapore, vol. 373, pp. 489–495, 2015.

[5] N. Farnaaz and M. A. Jabbar, "Random Forest Modeling for Network Intrusion Detection System," *Procedia Computer Science*, vol. 89, pp. 213–217, 2016.

[6] Z. Chen, L. Zhou and W. Yu, "ADASYN−Random Forest Based Intrusion Detection Model," in *2021 4th International Conference on Signal Processing and Machine Learning*, Beijing China: ACM, pp. 152–159, Aug. 2021.

[7] A. Thakkar and R. Lohiya, "A Review of the Advancement in Intrusion Detection Datasets," *Procedia Computer Science*, vol. 167, pp. 636–645, 2020.

[8] M. Khudadad and Z. Huang, "Intrusion Detection with Tree-Based Data Mining Classification Techniques by Using KDD," in *Machine Learning and Intelligent Communications*, Cham: Springer International Publishing, vol. 227, pp. 294–303, 2018.

[9] M. Belouch, S. El Hadaj and M. Idhammad, "Performance evaluation of intrusion detection based on machine learning using Apache Spark," *Procedia Computer Science*, vol. 127, pp. 1–6, 2018.

[10] N. Moustafa and J. Slay, "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Information Security Journal: A Global Perspective*, vol. 25, no. 1–3, pp. 18–31, Apr. 2016.

[11] W. L. Al-Yaseen, Z. A. Othman and M. Z. A. Nazri, "Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system," *Expert Systems with Applications*, vol. 67, pp. 296–303, Jan. 2017.

[12] Y. Xin *et al.*, "Machine Learning and Deep Learning Methods for Cybersecurity," in *IEEE Access*, vol. 6, pp. 35365-35381, 2018.

[13] R. Panigrahi and S. Borah, "A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems," *International Journal of Engineering & Technology*, vol.7, no.3.24, pp. 479-482, 2018.

[14] G. Louppe, 2014, "Understanding Random Forests: From Theory to Practice," arXiv, doi.org/10.48550/arXiv.1407.7502

[15] Y. Mirsky, T. Doitshman, Y. Elovici and A. Shabtai, 2018, "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection," arXiv, doi:10.48550/ARXIV.1802.09089.

[16] C. Zhang, F. Ruan, L. Yin, X. Chen, L. Zhai and F. Liu, "A Deep Learning Approach for Network Intrusion Detection Based on NSL-KDD Dataset," in *2019 IEEE 13th International Conference on Anti-counterfeiting, Security, and Identification (ASID)*, Xiamen, China: IEEE, pp. 41–45, Oct. 2019.

[17] S. M. Lundberg *et al.*, "From local explanations to global understanding with explainable AI for trees," *Nature Machine Intelligence*, vol. 2, no. 1, pp. 56–67, Jan. 2020.

[18] E. Yusuf Güven, S. Gülgün, C. Manav, B. Bakır and G. Zeynep Gürkaş Aydın, "Multiple Classification of Cyber Attacks Using Machine Learning," *Electrica*, vol. 22, no. 2, pp. 313–320, Jun. 2022.

[19] Canadian Institute for Cybersecurity, "IDS 2017 Datasets," University of New Brunswick. [Online]. Available: `https://www.unb.ca/cic/datasets/ids-2017.html`

[20] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321-357, Jun. 2002.

[21] X. -Y. Liu, J. Wu and Z. -H. Zhou, "Exploratory Undersampling for Class-Imbalance Learning,"

in *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 539-550, Apr. 2009.

[22] M. Lanvin, P.-F. Gimenez, Y. Han, F. Majorczyk, L. Mé, and É. Totel, "Errors in the CICIDS2017 Dataset and the Significant Differences in Detection Performances It Makes," in *Risks and Security of Internet and Systems*, Cham: Springer Nature Switzerland, vol. 13857, pp. 18–33, 2023.

[23] I. Sharafaldin, A. Habibi Lashkari and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization:," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, Funchal, Madeira, Portugal: SCITEPRESS - Science and Technology Publications, pp. 108–116, 2018.

[24] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat and S. Venkatraman, "Deep Learning Approach for Intelligent Intrusion Detection System," *IEEE Access*, vol. 7, pp. 41525–41550, 2019.

[25] Y. Zhou, G. Cheng, S. Jiang and M. Dai, "Building an efficient intrusion detection system based on feature selection and ensemble classifier," *Computer Networks*, vol. 174, p. 107247, Jun. 2020.

[26] Y. Li *et al.*, "Robust detection for network intrusion of industrial IoT based on multi-CNN fusion," *Measurement*, vol. 154, p. 107450, Mar. 2020.

Aomduan Jeamaon received her B.BA degree in Information Technology from Dhurakijbundit University, Thailand, in 2002, and her M.Science degree in Integrated Supply Chain Management from the same institution in 2011. Since 2003, Ms. Jeamaon has been working in the Financial Services industry, specializing in both Life and Non-life insurance sectors. Currently, she holds the position of Assistant Vice President in the Department of Business Development at Navakij Insurance Plc. Her responsibilities include planning and organizing marketing functions and operations, product development, products training, and providing systems technical support.

Her research interests encompass Cyber Security, Artificial Intelligence, and Data Science. Ms. Jeamaon has authored or co-authored two conference papers, focusing on "Cybersecurity Risk Assessment for Insurance in Thailand using Bayesian Network Model" and "A Classifiers Experimentation with Quantum Machine Learning."

With her extensive industry experience and academic background, Ms. Jeamaon brings a unique perspective to the intersection of technology and insurance, particularly in the areas of cybersecurity and advanced analytics.

Chaiyaporn Khemapatapan received his B.Eng. in Electrical Engineering from King Mongkut's Institute of Technology North Bangkok in 1994, M.Eng. in Electrical Engineering from Chulalongkorn University in 1997, and Ph.D. in Electrical Engineering from Chulalongkorn University in 2005. His doctoral thesis focused on "QPSK Impulse Signal Transmission for Ultra Wideband Bandwidth Communication Systems in the Presence of Multipath Channel." In 2004, he was a Research Student at the Mobile Communication Research Group (ARAKI LAB), Tokyo Institute of Technology, Japan, supported by the Thailand Research Fund (TRF) under the Royal Golden Jubilee (RGJ) Program.

Dr. Khemapatapan has an extensive career in academia spanning over two decades. Currently, he serves as the Dean of the College of Engineering and Technology and the Director of the Computer Engineering Graduate Program at Dhurakij Pundit University. His previous roles include Dean of the Faculty of Engineering (2012-2015) and various leadership positions within the university. His research expertise encompasses Computer Networking, Mobile and Wireless Networks, Cyber Security, Cloud Computing, and Machine Learning. Dr. Khemapatapan has authored and co-authored numerous international academic papers, with recent works focusing on transaction signing processes, multi-cloud storage security, DDoS attack defense in SDN, genetic algorithms, and smart grid distribution.

Dr. Khemapatapan's blend of academic excellence and administrative experience positions him as a key figure in engineering education and research in Thailand, particularly in the fields of computer engineering and telecommunications.