Developing an Intrusion Detection Framework for High-Speed Big Data Networks: A Comprehensive Approach

Kamran Siddique[1], Zahid Akhtar[2], Muhammad Ashfaq Khan[1], Yong-Hwan Jung[3], and Yangwoo Kim[1,*]
[1]Dongguk University, Seoul, South Korea
[2]University of Memphis, Memphis, USA
[3]Korea Institute of Science and Technology Information, Daejeon, South Korea
E-mail: [1]{kamran, ashfaq_jiskani, ywkim}@dongguk.edu, [2]zahid.eltc@gmail.com, [3]paul7931@kisti.re.kr
*Corresponding author: Yangwoo Kim

Received March 4, 2018; revised June 8, 2018; accepted July 13, 2018; published August 31, 2018

Abstract

In network intrusion detection research, two characteristics are generally considered vital to building efficient intrusion detection systems (IDSs): an optimal feature selection technique and robust classification schemes. However, the emergence of sophisticated network attacks and the advent of big data concepts in intrusion detection domains require two more significant aspects to be addressed: employing an appropriate big data computing framework and utilizing a contemporary dataset to deal with ongoing advancements. As such, we present a comprehensive approach to building an efficient IDS with the aim of strengthening academic anomaly detection research in real-world operational environments. The proposed system has the following four characteristics: (i) it performs optimal feature selection using information gain and branch-and-bound algorithms; (ii) it employs machine learning techniques for classification, namely, Logistic Regression, Naïve Bayes, and Random Forest; (iii) it introduces bulk synchronous parallel processing to handle the computational requirements of large-scale networks; and (iv) it utilizes a real-time contemporary dataset generated by the Information Security Centre of Excellence at the University of Brunswick (ISCX-UNB) to validate its efficacy. Experimental analysis shows the effectiveness of the proposed framework, which is able to achieve high accuracy, low computational cost, and reduced false alarms.

Keywords: Network intrusion detection systems, anomaly detection, bulk synchronous parallel, BSP, big data, machine learning, Darpa, KDD Cup 99, ISCX-UNB dataset

A preliminary version of this paper was presented at ICONI 2017, and was selected as an outstanding paper.
This research was supported by the MSIT (Ministry of Science, ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2018-2013-1-00717) supervised by the IITP (Institute for Information and Communications Technology Promotion). This research was also supported by Korea Institute of Science and Technology Information (KISTI).

http://doi.org/10.3837/tiis.2018.08.026 **ISSN : 1976-7277**

1. Introduction

Networked computer systems are increasingly becoming an integral part of today's information-overloaded modern society. The components which retain our society well-functioning and humming, with activities ranging from home shopping transactions to multi-billion dollar deals, are all dependent on large-scale networks. Nowadays, almost every facet of our daily lives has been significantly integrated with computing devices. Along with such technological advancements, a number of contemporary security threats in the digital world have also arisen, and therefore, protecting computer systems from various threats has become more concerning and important than ever before. Despite the availability of various security solutions, such as firewalls, access control systems, patch management, anti-virus, and anti-spyware applications, many computer systems are still vulnerable to security attacks that may inhibit their functioning, disclose private information, or create data corruption. Although these conventional security mechanisms appear as a first line of security defense, they are no longer sufficient to cope with the ever-evolving nature of intrusion skills and techniques. There is a pressing need to devise more efficient security solutions to make these systems tolerant and resistant to sophisticated network attacks [1], [2]. To this end, we have developed an intrusion detection framework, which serves as another line of security defense meant to mitigate or prevent network attacks.

The Internet is a global network of millions of interconnected computing devices that support the underlay for all computer-mediated activities. It is capable of transporting information that scales from a simple binary data to financial transactions and complex, real-time multimedia content without issue. Such ease and convenience is resulting with a massive increase in the number of Internet users. The Cisco Visual Networking Index has recently reported that the current global Internet protocol (IP) traffic is estimated to be 122 Exabytes (EB) per month and is expected to reach up to 278 EB per month by 2021 [3]. On the other side, a significant increase in the number of security attacks has also been noticed. In particular, the number of distributed denial-of-service (DDoS) attacks is expected to increase up to 3.1 million by 2021 [3]. The explosive growth and subsequent ubiquity of the Internet has naturally made networking systems the targets of enemies and criminals. The security of a computer system is compromised when it is illegitimately accessed by an individual or a program, possibly with plans to disrupt normal activities. In order to ensure security, intrusion detection systems (IDSs), especially network IDSs (NIDSs), are currently one of the most prominent solutions. A NIDS is often referred to as an anomaly-based intrusion detection system or simply an anomaly detection system. Notably, we use these terms interchangeably in this paper. A NIDS is a software or hardware component that aims to distinguish malicious actions, such as attempts to disrupt the confidentiality, integrity, or availability of a resource [4]. A NIDS possesses a significant value in the network security field and is considered a second security gate after a firewall. In recent years, network anomaly detection has become a major focus for network security researchers.

Intrusion detection techniques can primarily be classified into two categories based on whether the detection mechanism is signature-based (often called misuse-based) or anomaly-based. The working flow of both mechanisms is depicted in **Fig. 1**. Researchers are currently concentrating on anomaly-based intrusion detection system due to its ability to detect both known and unknown attacks.

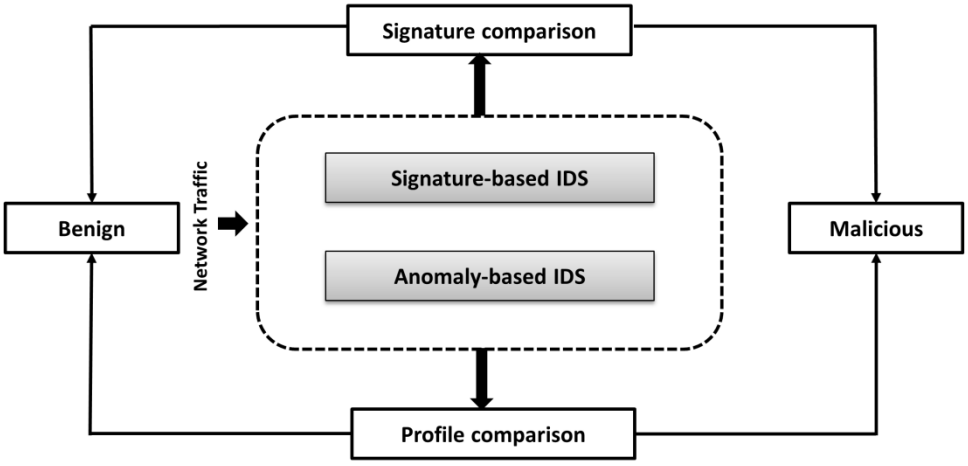

Fig. 1. Workflow of signature-based and anomaly-based IDSs

Over the past few years, anomaly detection based on machine learning techniques has received considerable attention from researchers. More emphasis has been given to devising efficient feature selection schemes and robust classification methods since they are generally considered the most vital characteristics for building an efficient IDS. Upon recognition of the advent of big data in the anomaly detection domain, researchers have started to deploy specialized big data frameworks and attempted to handle computational requirements efficiently [5], [6], [7]. Despite their great efforts, there are two significant factors that hinder the progress of anomaly detection research and desperately need the attention of the IDS research community. They are concerned with the decision to select an appropriate big data processing framework and to utilize appropriate datasets for the evaluation of an IDS. Most existing anomaly detection research has utilized the mapreduce paradigm and potentially outdated Darpa and KDD Cup 99 dataset families [8], [9]. Despite the remarkable qualities of mapreduce technology, it is not considered suitable to deploy for intensive iterative applications such as real-time traffic monitoring. Regarding performance evaluation of IDSs, datasets play a vital role. KDD Cup 99 datasets, the most valuable and innovative resource for anomaly detection research, were initially made available in 1998. The extensive use of the KDD dataset family, which is almost two decades old, in this modern era is depressing, particularly when superior alternatives are widely available [10], [11], [12], [13], [14]. These older and flawed dataset families lack big data veracity, modern footprint attacks, and have relatively poor quality. It is always vital to evaluate intrusion detection systems using appropriate datasets [15]. We therefore emphasize that the process of selecting appropriate computing technology and adequate datasets is an equally important and fundamental characteristic of developing a state-of-the-art IDS. To address the aforementioned challenges, we contribute to the literature by developing an efficient IDS that follows a comprehensive approach covering the following four aspects: optimal feature selection, robust classification scheme employment, utilization of appropriate big data framework, and usage of contemporary datasets to validate the effectiveness of the proposed system.

The rest of the paper is structured as follows. In the next section, we concisely present the background of network intrusion detection and related work. Section 3 presents the proposed

system and provides its architectural details. Section 4 presents the implementation and evaluation details followed by the conclusion in Section 5.

2. Background and Related Work

The idea of automatic intrusion detection was first carried out by James Anderson in 1980 in a classic paper [16] wherein he introduced a threat classification model to build a security monitoring surveillance system based on identifying malicious actions in user behavior.

An IDS diminishes the threat impact and handles such problems by performing a thorough analysis of the network traffic streams. It provides a more comprehensive defense against challenging threats and enhances network security. Cryptography and access control, on the other hand, are generally more focused on ensuring both confidentiality and integrity. In **Table 2**, we classify IDSs based on three important aspects: the environment they monitor, the employed detection approach, and their deployment architecture. We refer the reader to Axelsson Stefan [17] for a comprehensive resource on the taxonomy of IDSs.

Over the last 30 years, extensive research has been conducted to develop efficient NIDSs using various techniques, such as statistical methods, combination learners, and soft computing, as well as those based on knowledge, classification, and clustering [2], [18]. However, an exponential growth of massive data and advancements in networking domains has posed many challenges to researchers and practitioners in the field [19], [20]. Research efforts are underway to address such issues and challenges by devising techniques using big data frameworks, such as the Hadoop [21], Spark [22], and Storm [23] ecosystems. Recently, Manzoor et al. [24] proposed an intrusion detection system using support vector machine (SVM) to classify incoming network streams as benign or malicious. The authors utilized Apache Storm to handle the computational requirements for large-scale networks. It is an open source development platform generally used to build real-time big data stream processing applications. In this work, the proposed storm topology consists of one spout and three bolts: an input reader, the only spout which reads a network packet trace and forwards it to the next bolt, a data pre-processor, which is responsible to perform data conversion and normalization operations, an SVM algorithm, which performs the classification operations, and the result aggregator, which aggregates the classification results and stores them in a file. The authors utilized KDD Cup 99 datasets to perform system validation; however, a detailed analysis of experimental protocol and some important performance metrics are also missing in this work.

Kang and Kim [25] proposed a wrapper-based feature selection method to detect network anomalies. The authors focused on detection of denial of service attacks in this paper. The main idea is based on utilizing the problem of combinatorial optimization and an optimal feature selection algorithm. The proposed feature selection algorithm works similar to the well-known meta-heuristic algorithms that are widely used to implement combinatorial optimization problems. Generally, the accuracy of the final classifier in wrapper-based feature selection is used as a cost function during the search process; however, the proposed system adopted the approach of clustering accuracy over the training dataset. The k-means clustering algorithm has been used to group the training dataset. In order to validate the performance of the proposed system, a multi-layer perceptron has been implemented and the overall system validation has been performed using NSL_KDD dataset. The proposed system achieved considerable detection accuracy; however, it suffers with the problem of the false alarm rate.

Table 2. Classification of Intrusion Detection Systems

Classification Aspect	IDS Type	Description
Monitoring environment	Host-based (HIDS)	• Runs on individual hosts or devices on the network. • Monitors the inbound and outbound network streams from the system only and will alert the user if suspicious activity is detected.
	NIDS	• Attempts to identify unauthorized, illicit, and anomalous activities based solely on network traffic.
	Hybrid	• An IDS that utilizes the functions of both HIDS and NIDS.
Detection approach	Signature-based	• Refers to the detection of network attacks by looking for specific data patterns, such as byte sequences in network streams, or known malicious instruction sequences used by malware. • This terminology originates from anti-virus applications, which refer to these detected patterns as signatures. • Although signature-based IDSs can easily detect known attacks, it is impossible to detect unknown or new attacks, for which no pattern is available.
	Anomaly-based	• Primarily introduced to detect unknown attacks, in part due to the rapid development of malware. • The basic approach is to use machine learning to create a model of trustworthy activity and then compare new behavior against this model. • Although this approach enables the detection of previously unknown attacks, it generally suffers with generating false alarms; previously unknown legitimate action may also be classified as malicious.
	Hybrid	• A system that exploits benefits of both HIDS and NIDS. • Attempts to detect known as well as unknown attacks.
Deployment architecture	Distributed	• A distributed IDS consists of several multiple intrusion detection subsystems over a large-scale network, all of which communicate with each other. • In addition to its basic functionality, it communicates to exchange attack alerts data that can be configured to operate in a distributed manner, such as an open source system OSSEC.
	Non-distributed	• An IDS that can be deployed only at a single location such as an open source system Snort.

Rathore et al. [7] proposed a real-time intrusion detection framework for ultra-high-speed big data environment using the Hadoop framework. The architecture of the proposed IDS consists of four layers: traffic capturing, which reads the network traffic; a filtration and load balancing server, which performs filtration of the network traces to achieve preliminary flow identification using the in-memory database and the load balancing of network traces among master and slave nodes using IP addresses; and the processing layer, which is an important part of the system composed of various master and data nodes of Hadoop. The authors also briefly claimed to have used Apache Spark to attain real-time processing capabilities, but this usage has not been justified well. There is also the decision server layer, which performs classification tasks using machine learning algorithms such as REPTree, J48, and SVM. The system validations were performed using a collection of DARPA, KDD Cup 99, and NSL_KDD datasets.

There are some studies that have utilized the ISCX-UNB dataset to conduct appropriate system validation. However, there are still immense demands necessary to make several improvements, such as improving accuracy and reducing false alarms [26], [27], [28], [29], [30], [31]. Moreover, most studies do not address the important characteristics of NIDS research comprehensively, such as specialized feature selection techniques, robust and appropriate machine learning classifiers, and handling big data issues in high-speed networks, to name a few. We present a comparative outlook of our work with these studies in Section 4.

3. Proposed Framework

The architecture of the proposed IDS framework is depicted in **Fig. 2**. Fundamentally, it involves the analysis of network traffic and is compared with the defined baseline, which shows the normal behavior of the system on all matching operations. If a mismatch is found, the system generates an alert indicating the occurrence of malicious activity. The functionality of the proposed system is as follows.

3.1 Input

The most fundamental and significant inputs to IDS are network flows in real-time environments and recorded network traces, often called datasets or workloads. The type, quality, and location where data is collected are the determinant factors in the design and effectiveness of an IDS. We believe that the productivity of intrusion detection research is largely dependent on the quality of datasets being used in addition to the computational techniques involved. Therefore, we decided to use the ISCX-UNB dataset [10] as the input of our proposed system for further experimental evaluations. The details of the dataset are given in Section 4.

3.2 Analysis

This is the core of the proposed framework. It performs an in-depth analysis of the network traffic and involves several components, such as data preprocessing, feature ranking and selection, BSP-based machine learning classifiers, and attack recognition.

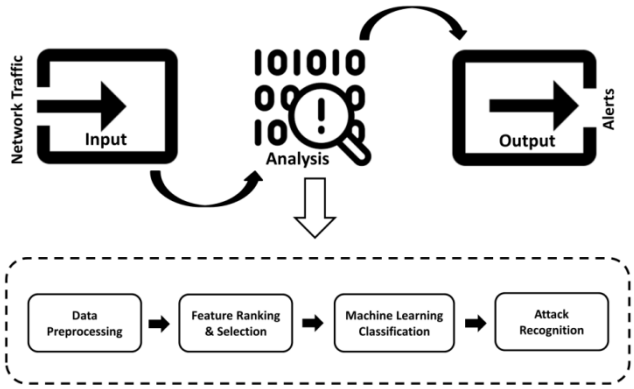

Fig. 2. The architecture of the proposed intrusion detection framework

The data preprocessing component is responsible for preprocessing the data involving conversion and normalization operations to yield the data for feature ranking as well as selection steps. The dataset is composed of assorted types of data, including symbolic and numeric representation such as *protocolName* and *totalDestinationpackets*, respectively. Since classification approaches needs each input data record in the form of real number vectors, each symbolic feature is first transformed into a numerical value in this phase. The integers from 0 to N – 1 are assigned to each symbolic feature and then each value is linearly scaled for the range of [0–1]. After conversion, in order to avoid the biasing factor of features with higher values, data normalization is performed. Feature selection in intrusion detection is used to eliminate redundant and insignificant data by choosing a subset of features from the original available features. The objective is to reduce the feature space according to certain criterion to improve the predictive accuracy of classification algorithms. To accomplish this task, we first applied the information gain (IG) measure that ranks features using their importance, which is followed by the automated branch-and-bound (ABB) technique to obtain the optimal feature subset [32]. Information gain evaluates the worth of an attribute as a measure with respect to class. It has likelihood of selecting features with high distinguishing values and works using the entropy principle, which has been widely applied in the information theory domain. In our experiments, we calculate information gain for each class attribute using the following probability definitions.

$$P(c_i|X) = \frac{\mathrm{P(c_i)P(X|c_i)}}{\mathrm{P(X)}}, \tag{1}$$

$$P(X) = \sum P(c_i)P(X|c_i) \tag{2}$$

where $P(c_i)$ defines the prior probabilities for all classes i and $P(\mathrm{X}|c_i)$ refers to the conditional probabilities of X in class c_i.

If there are d numbers of classes c_i in the data D, and X and p denote the feature and partitions, respectively, information gain is obtained using the series of following equations.

$$I(D) = -\sum_{i=1}^{d} P_D\,(c_i) log_2 P_D(c_i), \tag{3}$$

where the information for Dj owing to partition D at X is estimated as

$$I\left(D_j^X\right) = -\sum_{i=1}^{d} P_{D_j^X}(c_i) log_2 P_{D_j^X}(c_i), \quad (4)$$

and for the feature X, the information gain is computed as follows:

$$IG(X) = I(D) - \sum_{j=1}^{p} \frac{|D_j|}{|D|} I\left(D_j^X\right), \quad (5)$$

where $|D|$ is the number of instances in D and $P_D(c_i)$ represent the prior probabilities for the data D estimated by $P(c_i) = |c_{i,D}|/|D|$.

The process of IG evaluation and feature selection using the ABB technique is listed in Algorithms 1 and 2, respectively. There are a total of 18 features in the ISCX-UNB dataset and Algorithm 1 was first used to rank the features prior to selecting the optimal set of features. **Table 2** shows the details of all features along with the values of the corresponding IG measure in a decreasing order while **Fig. 3** gives a graphical representation of the IG measures in the order given by the original dataset.

Algorithm 1: Feature ranking algorithm using information gain

```
Input: D – A train dataset with all features Xi, i=1,2,3…N
Output: Features ranked by information gain
1: initialize empty list L
2: do
3:   compute IG(Xi);   /* via equation (5) */
4:   add Xi with L in descending order w.r.t. IG(Xi);
5: while (i=1 to N);
6: Return L
```

Algorithm 2: ABB feature selection algorithm

```
Input: S – A training dataset D with all features Xi
where i=1,2,3…N
        Q – An empty queue, S1, S2 – temporary subsets
        U – Evaluation measure (inconsistency)
Output: A selected feature subset S
1: initialize L = {S}
2: α = U(S2, D)
3: ABB(S, D)  /* Automated branch-and-bound function
4:                 reading all features in data D */
5: do
6:   S1 = S – X;
7:   add S1 to Q;
8:   while Q is NOT empty
9:     S2 = delete from Q;
10:     if (S2 is valid & U(S2, D) <= α)
11:      Append S2 in L;
12:    ABB(S2, D)
13: while (i=1 to N);
14: Return the minimum subset;
```

Table 2. The information gain measure for all features

S. No.	Feature Rank	Feature Name	Information Gain
1	f4	totalDestinationPackets	0.928912
2	f5	totalSourcePackets	0.822597
3	f9	Direction	0.807751
4	f12	Source	0.782945
5	f13	protocolName	0.581982
6	f15	Destination	0.441058
7	f17	startDateTime	0.422889
8	f18	stopDateTime	0.365589
9	F16	destinationPort	0.306328
10	F14	sourcePort	0.303309
11	f3	totalDestinationBytes	0.297390
12	f2	totalSourceBytes	0.278352
13	f7	destinationPayloadAsBase64	0.269263
14	f6	sourcePayloadAsBase64	0.060201
15	f8	destinationPayloadAsUTF	0.055125
16	f11	destinationTCPFlagsDescription	0.050177
17	f10	sourceTCPFlagsDescription	0.041616
18	f1	appName	0.024322

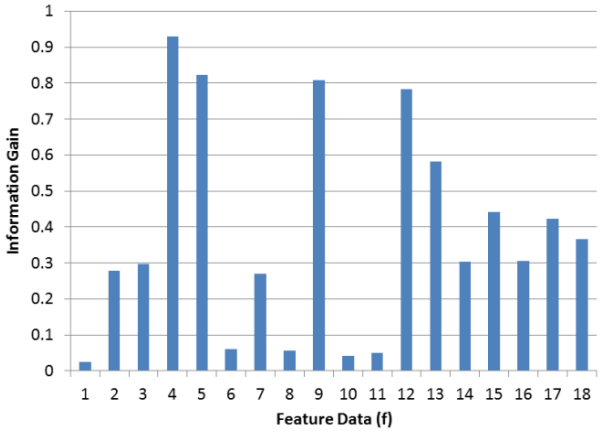

Fig. 3. Information gain measure for each feature

Once the features are ranked, the ABB algorithm is employed to find the optimal feature subset using the inconsistency rate U as an evaluation measure. The process for calculating U over a given dataset D may be defined in a series of three steps. Step I: two instances of the feature set are considered to be inconsistent if they have the same values but different class labels; for example, if the two instances have the values 011100001 and 011100000, they represent the same data for all attributes except their class labels (i.e., 1 and 0). Step II: subtracting the largest number of instances of different class labels yields inconsistency count from the total number of matching instances. For example, if there are n matching instances and C_1 and C_2 denote the number of instances for given class labels, then if C_2 is greater than C_1, the inconsistency count will be n - C_2 or vice versa. Step III: ratio between 'sum of inconsistency counts' and 'total number of instances' produces the inconsistency rate. The irrelevant and redundant features can be removed effectively by using inconsistency as an

evaluation measure in an ABB algorithm. Contrary to branch-and-bound conventional algorithms, ABB estimates bounds automatically. Given a set of features, this algorithm removes one feature at a time via a breath-first search technique until it reaches the base criteria. The process is as follows. Each node acts as a subset of features for a legitimacy test to certify that an execution is valid such that if 'Hamming distance (node being visited, a pruned node) ~= 1', then it is legitimate. Under each iteration, one feature, whose time complexity is $O(N)$, is dropped; here, N is the number of features. Therefore, to complete the process, m iterations are required, which leads to an overall time complexity of ABB to $O(mN)$. The maximum number of children m that a node can have is always smaller than N. The combined outcomes of Algorithms 1 and 2 yield the best feature subset for classification mechanisms, namely, totalDestinationPackets, totalSourcePackets, direction, source, protocolName, destination, startDateTime, and stopDateTime.

It is well-documented in machine learning literature that the choice of classification schemes majorly affects the overall performance of the system even if an optimal set of features is used. During the testing phase, classification is a process in which the system predicts true class label(s) for a given unseen set of features. Various algorithms have been proposed to devise highly effective anomaly detection systems. It is worth noting that the incorrect selection of a classification algorithm may cause high false alarms and high computational costs [15]. The optimal choice of classification is still an open issue in the intrusion detection domain. The tactful amalgamation of feature extraction techniques and classification schemes may produce better IDS. To this aim, in this study, we implemented Logistic Regression (LR), Naïve Bayes (NB), and Random Forest (RF) techniques to classify network traffic [33]. They have been chosen based on the overall performance of the system. Moreover, they are both pliable to implement and capable of updating their execution approaches by incorporating new information. Once an optimal subset of features is obtained, it is used as the input for the classifier-training phase, where we employed three efficient machine-learning classification techniques, as given below.

Logistic is a ridge estimator based classifier that is well-suited to deploy for two-class classification problems [34]. The relationship between binary outcomes and independent variables is calculated using a binary logistic model's probability. Specifically, the existence (1) or nonexistence (0) of a specific attribute or outcome in general is described by binary outcomes, which find the existence of a specific activity y in a given feature set x. For instance, y is assigned a value of 1 if a certain activity is present and is otherwise 0. For the anomaly detection system, the network traffic flow whether malicious or normal is depicted by the value of y. In anomaly detection, every record/flow is given a probability that is then utilized to decide whether it is anomalous or benign. A logistic curve is generated by the model such that posterior probabilities lie between 0 and 1, which causes simple linear regression schemes to be ineffective since they have inherent attribute of allowing the dependent variable to proceed these limits and produce inconsistent results. During the training phase, the classifier is trained using the selected subset of features on a training database. The test data is fed to the stored trained model to detect intrusions. The trace matching operation treats normal class as normal data, whereas others as intrusive attacks.

The NB classifier is a well-known supervised learning technique for classification problems [35]. It has shown efficacy in diverse fields of applications ranging from image classification to disease predictions [36], [37]. NB depends on Bayesian theorem, which is very practical for high dimensionality inputs and large datasets. In spite of its simplicity, an NB classification scheme can usually outperform more sophisticated classification algorithms.

It is a construction classifier, an algorithm that is used to attribute class labels to problem instances. The main assumption of the NB classification method is independence; it assumes that all features are independent and that the presence of a feature in a class has no relation with the presence of any other feature in same class such that correlations between all considered features in a given study are unrelated or ignored.

The RF classification scheme is an ensemble learning method based on decision trees, which are used both for classification and regression [38], [39], [40]. RFs are a combination of tree predictors where every tree relies on estimations of a random vector tested autonomously. The same distribution is applied to all trees in the forest. Compared to other bagging techniques, in RF, at every fundamental classification tree hub, an arbitrary indicator factors subset is employed as the variable to determine the split. During the training process, RF makes multiple classifications and regression trees prepared using a bootstrapped test of training samples and inquiries using only a randomly selected subset of the input variables to estimate a split for every node. Gini index of node impurity is utilized for computing splits in the predictor variables. During the testing/classification process, every node/tree gives a unit vote on the most popular class for the input. The majority vote is used to obtain the outcome of the classifier. RF is generally selected over other tree-based methods because it is very effective with noise and does not over-fit. Since the trees in RF are not pruned, the computational many-sided quality is decreased, which allows RF to deal with high dimensional features and data using only a considerable number of trees in the ensemble.

Once the classifiers are trained using the selected subset of features, the stored trained classifier can then be employed to detect normal and intrusive data. The test data is then transported to the saved trained model to detect intrusions. The traces matching the normal class are treated as normal data while the others are reported as intrusive activities.

3.3 Output

The final phase of the presented framework is similar to the output component of the traditional system. It is responsible for presenting the processed data in a usable format. In other words, it largely interfaces with the user and generates alerts.

4. Experimental Evaluation

Considering the significance of using appropriate datasets to evaluate IDSs, as mentioned above in this paper, we utilized ISCX-UNB datasets [10] rather than following the traditional approach and using the legacy KDD dataset family. The dataset was collected in a week with practical and systematic situations reflecting network traffic and intrusions. Specifically, the dataset is labeled for normal and malicious flows for a total of 2,381,532 and 68,792 records in each respective category. Additionally, a variety of multi-stage attack scenarios were performed to produce malicious traces (e.g., infiltration from the inside, HTTP, DoS, DDoS via an Internet Relay Chat (IRC) botnet, and brute force Secure Shell (SSH)). The train and test dataset distributions utilized in this study is presented in **Table 3**.

The experiments were performed on an Intel core i7-6500U CPU @2.5 GHz with a 512 GB SSD and 8 GB RAM with Apache Hama [41], [42] version 0.7.1 installed on Ubuntu 12.04. To demonstrate the efficacy of presented framework in this study, we used standard performance metrics namely, accuracy, detection rate (DR), and false positive rate (FPR) [43]. The overall performance of the system is promising, as shown in **Tables 4 and 5** and **Fig. 4**.

Table 3. Distribution of training and testing datasets

Dataset/Network Flows	No. of Features	Training		Testing	
		Benign	Malicious	Benign	Malicious
ISCX-UNB-SAT	8	85,222	1,353	45,889	1,353
ISCX-UNB-MON	8	108,945	2,451	58,664	1,320
ISCX-UNB-TUE	8	347,308	24,295	187,012	13,083
ISCX-UNB-WED	8	339,470	0	182,793	0
ISCX-UNB-THU	8	255,054	3,381	137,338	1,822

Table 4. Performance results of the proposed framework

Network Flows	LR			NB			RF		
	DR	FPR	Accuracy	DR	FPR	Accuracy	DR	FPR	Accuracy
ISCX-UNB-SAT	98.09	0.18	99.15	97.25	0.71	98.12	99.21	0.12	99.65
ISCX-UNB-MON	99.39	0.58	99.53	98.50	0.28	97.95	98.86	0.31	99.38
ISCX-UNB-TUE	98.56	0.67	98.99	95.96	0.54	97.86	99.45	0.34	99.71
ISCX-UNB-WED	99.11	0.45	99.26	92.31	0.41	96.45	99.62	0.51	99.72
ISCX-UNB-THU	99.23	0.39	99.44	94.28	0.34	97.43	99.43	0.20	99.69

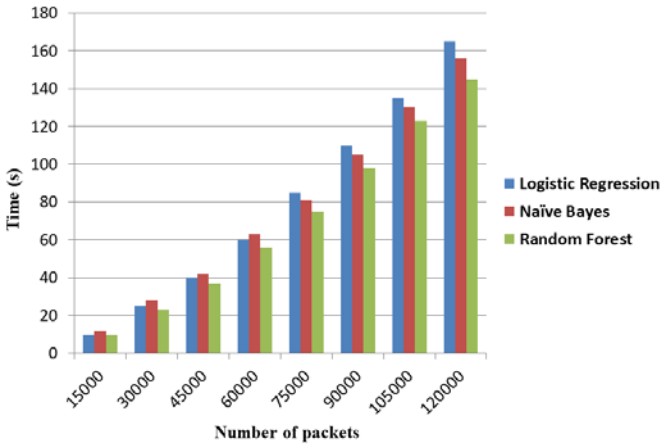

Fig. 4. Training time taken by the machine learning classifiers

Several observations may be extracted from **Table 4** and **Fig. 4**. First, the proposed framework presents high potential as a simple, unconstrained, implicit method for anomaly detection with great recognition accuracy for diverse, practical in-the-wild network flow traffics. Second, the error rates decrease as number of packets goes from low to high. Third, it is evident that the RF classifier performs better than both LR and NB because RF is capable of decreasing variances and norming out the biases and most unlikely overfitting. Finally, LR

performed in DR was accurate compared to NB owing to LR's one of main features, which is that the independent variables do not necessarily have to be normally distributed, while NB assumes that all attributes are independent (i.e., no correlation between variables), and it is well-documented that correlation mapping is beneficial for attaining better accuracy. In addition, NB also assumes that the samples follow a Gaussian distribution, which is usually true for small datasets. However, network intrusion detection databases are heterogeneous and contain different attack types and sizes, and thereby NB either is overfitted or could not handle the concept-drift issue. Regarding the time efficiency of the proposed system, we used training time as an evaluation metric in reference to the time taken to build the training model. **Fig. 4** illustrates the time taken by each machine learning classifier for a range of network packets. We can notice that the time taken by each classifier is promising; however, RF outperforms others, achieving 143 s for 120,000 network packets. Hence, the presented scheme is capable in respect of low computational cost in addition to assuring attack detection accuracy.

Table 5 compares our results with existing solutions for the ISCX-UNB dataset. This dataset was generated much later than the DARPA and KDD dataset family, so there are relatively fewer corresponding experimental results available [9]. Based on the available evaluation results for the compared methods, the best results for each study have been selected in terms of accuracy and false alarms. It is easy to see that the proposed system performs better both in terms of accuracy and false alarms compared to state-of-the-art methods. This is mainly because of the efficient feature selection technique we used and the implementation of appropriate machine learning classifiers. It is worth noticing that the comparisons are for reference only as many researchers have used different proportions of traffic types and dataset distributions, preprocessing techniques, and sampling methods. Therefore, a straightforward comparison for some metrics, such as training and testing time, is generally not considered suitable [43]. Although our approach achieved better performance for the considered evaluation metrics, it cannot be claimed that the proposed solution completely outperformed other methods. Nevertheless, we assert that one can obtain a noteworthy level of security and convenience against intrusion attacks using the proposed technique, which is simple, fast, effective, and highly suitable for real-time applications as well.

Table 5. Comparison with existing solutions using ISCX-UNB dataset

Authors	Algorithm(s) or Technique	Accuracy or DR (%)	False Alarms (%)
Kakavand et al. [26]	PCA	97.0	1.2
Kumar et al. [27]	AMGA2-NB	94.5	7.0
Yassin et al. [28]	KMC+NBC	99.0	2.2
Tahir et al. [29]	KMC-D+NBC	99.5	1.2
Tan et al. [30]	MCA+EMD	90.12	7.92
Sally et al. [31]	PLL+NGL	95.31	0.80
Proposed solution	RFF+ABB	99.72	0.51

5. Conclusion

In this paper, we followed a comprehensive approach to develop an efficient IDS with emphasis on tackling big data problems in large-scale networks. In order to cope with the challenges, which are mainly caused by the volume, velocity, variety, and veracity of the

workloads, the proposed system incorporated a powerful bulk synchronous parallel computing engine that was capable of handling a large volume of network traffic in real-time environments. Our approach is also novel in utilizing the ABB technique to select optimal feature subsets and performing classification tasks using efficient machine learning techniques. We also highlighted the role and significance of using appropriate datasets, which is greatly lacking in existing studies. The experimental results for the contemporary dataset verify the accuracy and efficiency of the proposed system.

We believe that the proposed approach can also be customized for other domains, such as anomaly detection in image data, streaming anomaly detection, and detecting intrusions in time series data, since these have very peculiar characteristics and our proposed framework can be a suitable solution with some customization. This work could also be extended to devising more novel NIDSs solutions for more generalized systems and to develop cross-dataset methods, which have not received much attention in the field of network anomaly detection, i.e., a process in which a system is being trained on one dataset and tested on another. This may provide an additional method of evaluating the interoperability and generalization capability of an IDS.

References

[1] K. Grahn, M. Westerlund, and G. Pulkkis, "Analytics for network security: A survey and taxonomy," in *Proc. of Information Fusion for Cyber-security Analytics*, Springer, New York, NY, USA, pp. 175-193, 2017. Article (CrossRef Link)

[2] A. L. Buczak, and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, Vol. 18, No. 2, pp. 1153-1176, 2016. Article (CrossRef Link)

[3] Cisco Visual Networking Index, The Zettabyte Era: Trends and Analysis, June 2017.

[4] R. Heady, G. F. Luger, A. Maccabe, and M. Servilla, "The architecture of a network level intrusion detection system," *Technical Report, Department of Computer Science. College of Engineering, University of New Mexico*, Albuquerque, NM, USA, 15 August 1990. Article (CrossRef Link)

[5] V. P. Janeja, A. Azari, J. M. Namayanja, and B. Heilig, "B-dids: Mining anomalies in a Big-distributed Intrusion Detection System," in *Proc. of Proceedings of the 2014 IEEE International Conference on Big Data (Big Data)*, Washington, DC, USA, 27–30 October 2014, pp. 32–34. Article (CrossRef Link)

[6] R. Kumari, M. K. Singh, R. Jha, and N. K. Singh, "Anomaly detection in network traffic using K-mean clustering," in *Proc. of IEEE 3rd International Conference on Recent Advances in Information Technology (RAIT)*, Dhanbad, India, 3-5 March 2016, pp. 387-393. Article (CrossRef Link)

[7] M. M. Rathore, A. Ahmad, and A. Paul, "Real time intrusion detection system for ultra-high-speed big data environments," *The Journal of Supercomputing*, Vol. 72, No. 9, pp. 3489-3510, 2016. Article (CrossRef Link)

[8] R. Zuech, T. M. Khoshgoftaar, and R. Wald, "Intrusion detection and Big Heterogeneous Data: a survey," *Journal of Big Data*, Vol. 2, No. 1, 2015. Article (CrossRef Link)

[9] A. Özgür, and H. Erdem, "A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015," *PeerJ PrePrints*, 2016. Article (CrossRef Link)

[10] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers & Security*, Vol. 31, No. 3, pp. 357-374, 2012. Article (CrossRef Link)

[11] MAWI Working Group Traffic Archive: Available online: Article (CrossRef Link) (accessed on 20 February 2018).

[12] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Towards Generating Real-life Datasets for Network Intrusion Detection," *IJ Network* Security, Vol. 17, No. 6, pp. 683-701, 2015.
[13] The UNSW-NB15 Dataset: Available online: Article (CrossRef Link) (accessed on 20 February 2018).
[14] W. Haider, J. Hu, J. Slay, B. P. Turnbull, and Y. Xie, "Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling," *Journal of Network and Computer Applications*, Vol. 87, pp. 185-192, 2017. Article (CrossRef Link)
[15] R. Sommer, and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *Proc. of IEEE Symposium on Security and Privacy (SP)*, pp. 305-316, 2010. Article (CrossRef Link)
[16] J. P. Anderson, "Computer security threat monitoring and surveillance," Technical Report, Vol. 17, Fort Washington, USA, 1980.
[17] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Technical Report, Vol. 99, 2000.
[18] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: methods, systems and tools," *IEEE Communications Surveys & Tutorials*, Vol. 16, No. 1, pp. 303-336, 2014. Article (CrossRef Link)
[19] S. Suthaharan, "Big data classification: Problems and challenges in network intrusion prediction with machine learning," *ACM SIGMETRICS Performance Evaluation Review*, Vol. 41, No. 4, pp. 70-73, 2014. Article (CrossRef Link)
[20] L. Cheng, F. Liu, and D. D. Yao, "Enterprise data breach: causes, challenges, prevention, and future directions," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol. 7, No. 5, 2017. Article (CrossRef Link)
[21] Apache Hadoop. Available online: Article (CrossRef Link) (accessed on 20 February 2018).
[22] Apache Spark. Available online: Article (CrossRef Link) (accessed on 20 February 2018).
[23] Apache Storm. Available online: Article (CrossRef Link)(accessed on 20 February 2018).
[24] M. A. Manzoor, and Y. Morgan, "Network intrusion detection system using apache storm," *Advances in Science, Technology and Engineering Systems Journal*, Vol. 2, Issue 3, pp. 812-818, 2017. Article (CrossRef Link)
[25] S. H. Kang, and K. J. Kim, "A feature selection approach to find optimal feature subsets for the network intrusion detection system," *Cluster Computing*, Vol. 19, No. 1, pp. 325-333, 2016. Article (CrossRef Link)
[26] M. Kakavand, N. Mustapha, A. Mustapha, and M. T. Abdullah, "Effective Dimensionality Reduction of Payload-Based Anomaly Detection in TMAD Model for HTTP Payload," *KSII Transactions on Internet and Information Systems*, Vol. 10, No. 8, pp. 3884-3910, 2016. Article (CrossRef Link)
[27] G. Kumar, and K. Kumar, "Design of an evolutionary approach for intrusion detection," *The Scientific World Journal*, 2013. Article (CrossRef Link)
[28] W. Yassin, N. I. Udzir, Z. Muda, and M. N. Sulaiman, "Anomaly-based intrusion detection through k-means clustering and naives bayes classification," in *Proc. of Proceedings of 4th International Conference on Computing and Informatics* (*ICOCI)*, No. 49, pp. 298-303, 2013.
[29] M. H. Tahir, A. M. Said, N. H. Osman, N. H. Zakaria, P. N. M. Sabri, and N. Katuk, "Oving K-Means Clustering using discretization technique in Network Intrusion Detection System," in *Proc. of IEEE 3rd International Conference on Computer and Information Sciences (ICCOINS)*, 15-17 August 2016, Kuala Lampur, Malaysia, pp. 248-252. Article (CrossRef Link)
[30] Z. Tan, A. Jamdagni, X. He, P. Nanda, R. P. Liu, and J. Hu, "Detection of denial-of-service attacks based on computer vision techniques," *IEEE Transactions on Computers*, Vol. 64, No. 9, pp. 2519-2533, 2015. Article (CrossRef Link),
[31] H. Sallay, A. Ammar, M. B. Saad, and S. Bourouis, "A real time adaptive intrusion detection alert classifier for high speed networks," in *Proc. of IEEE 12th International Symposium on Network Computing and Applications (NCA)*, 22-24 August 2013, Cambridge, MA, USA, pp. 73-80. Article (CrossRef Link)

[32] H. Liu, and H. Motoda, "Data reduction via instance selection," *Instance selection and construction for data mining*, pp. 3-20. Springer, Boston, MA, 2001. Article (CrossRef Link)
[33] H. Trevor, T. Robert, and J. Friedman, "The elements of statistical learning," Vol. 1, 2001.
[34] Jr. D.W. Hosmer, S. Lemeshow, and R. X. Sturdivant, "Applied logistic regression," Vol. 398, *John Wiley & Sons*, Hoboken, NJ, USA, 2013.
[35] I. Rish, "An empirical study of the naive Bayes classifier," in *Proc. of IBM IJCAI Workshop on Empirical Methods in Artificial Intelligence*, Vol. 3, No. 22, pp. 41-46, 2001.
[36] S. McCann, and D. G. Lowe, "Local naive bayes nearest neighbor for image classification," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 16-21 June 2012, Providence, RI, USA, pp. 3650-3656. Article (CrossRef Link)
[37] M. Langarizadeh, and F. Moghbeli, "Applying naive Bayesian networks to disease prediction: a systematic review," *Acta Informatica Medica*, Vol. 24, No. 5, 2016. Article (CrossRef Link)
[38] G. Biau, "Analysis of a random forests model," *The Journal of Machine Learning Research*, pp. 1063-1095, 2012.
[39] K. Fawagreh, M. M. Gaber, and E. Elyan, "Random forests: from early developments to recent advancements," *Systems Science & Control Engineering: An Open Access Journal*, Vol. 2, No. 1, pp. 602-609, 2014. Article (CrossRef Link)
[40] M. Denil, D. Matheson, and N. D. Freitas, "Narrowing the gap: Random forests in theory and in practice," in *Proc. of International Conference on Machine Learning (ICML)*, 2014.
[41] K. Siddique, Z. Akhtar, E. J. Yoon, Y. S. Jeong, D. Dasgupta, and Y. Kim, "Apache Hama: an emerging bulk synchronous parallel computing framework for big data applications," *IEEE Access*, Vol. 4, pp. 8879-8887, 2016. Article (CrossRef Link)
[42] K. Siddique, Z. Akhtar, Y. Kim, Y. S. Jeong, and E. J. Yoon, "Investigating Apache Hama: a bulk synchronous parallel computing framework," *The Journal of Supercomputing*, Vol. 73, No. 9, pp. 4190-4205, 2017. Article (CrossRef Link)
[43] M. Sokolova, and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, Vol. 45, No. 4, pp. 427-437, 2009. Article (CrossRef Link)

Kamran Siddique is a research assistant professor at Dongguk University, Seoul, South Korea. His research interests include cyber security, machine learning, and big data processing. Siddique received a PhD in computer engineering from Dongguk University, Seoul, South Korea.

Zahid Akhtar is a research assistant professor at University of Memphis, Memphis, USA. Prior to his current role, he has been a postdoctoral researcher at the INRS-EMT center, University of Quebec, Canada, Bahcesehir University, Turkey and University of Udine, Italy. His research interests include computer vision, pattern recognition, affective computing, security systems, and big data applications. He is a member of the IEEE Signal Processing Society. Akhtar received a PhD in electronic and computer engineering from University of Cagliari, Italy.

Muhammad Ashfaq khan is a PhD student at Dongguk University, Seoul, South Korea. His research interests include big data analytics, artificial intelligence, machine learning, deep learning, cloud computing, and computer networks. Khan received the Master of Engineering degree in communication systems & networks from Mehran University of Engineering & Technology, Jamshoro, Pakistan.

Yong-Hwan Jung is a senior researcher of Supercomputing Service Center at Korea Institute of Science and Technology Information. His research interests include high performance computing, cloud computing, network security and software defined networking. Jung received the MS degree in computer science from Soongsil University, Seoul, South Korea.

Yangwoo Kim (Corresponding author) is a professor at Dongguk University, Seoul, South Korea and an International Standard Expert on cloud computing, ISO/IEC JTC-1, and ITU-T. His research interests include parallel and distributed processing systems, cloud computing, big data, grid computing, and P2P computing. Kim received a PhD in computer engineering from Syracuse University, New York, USA.