IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON ARTIFICIAL INTELLIGENCE (AI)-EMPOWERED INTELLIGENT TRANSPORTATION SYSTEMS

Received September 26, 2019, accepted October 7, 2019, date of publication October 18, 2019, date of current version November 4, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2948382

A Distributed Network Intrusion Detection System for Distributed Denial of Service Attacks in Vehicular Ad Hoc Network

YING GAO, HONGRUI WU, BINJIE SONG, YAQIA JIN, XIONGWEN LUO, AND XING ZENG
Department of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

Corresponding authors: Hongrui Wu (skyhongrui@163.com) and Binjie Song (sbj332644514@163.com)

This work was supported in part by the Natural Science Foundation of Guangdong Province under Grant 2018A030313351, in part by the Key Research and Development Program of Guangdong Province under Grant 2019B010137001, in part by the Internet Innovation and Development Project, in 2018 under Grant MIZ1824020-2, and in part by the Key Laboratory of Information Network Security, Ministry of Public Security.

ABSTRACT Security assurance in Vehicular Ad hoc Network (VANET) is a crucial and challenging task due to the open-access medium. One great threat to VANETs is Distributed Denial-of-Service (DDoS) attack because the target of this attack is to prevent authorized nodes from accessing the services. To provide high availability of VANETs, a scalable, reliable and robust network intrusion detection system should be developed to efficiently mitigate DDoS. However, big data from VANETs poses serious challenges to DDoS attack detection since the detection system require scalable methods to capture, store and process the big data. To overcome these challenges, this paper proposes a distributed DDoS network intrusion detection system based on big data technology. The proposed detection system consists of two main components: real-time network traffic collection module and network traffic detection module. To build our proposed system, we use Spark to speed up data processing and use HDFS to store massive suspicious attacks. In the network collection module, micro-batch data processing model is used to improve the real-time performance of traffic feature collection. In the traffic detection module, the classification algorithm based on Random Forest (RF) is adopted. In order to evaluate the accuracy of detection, the algorithm was evaluated and compared in the datasets, containing NSL-KDD and UNSW-NB15. The experimental results show that the proposed detection algorithm reached the accuracy rate of 99.95% and 98.75%, and the false alarm rate (FAR) of 0.05% and 1.08%, respectively, in two datasets.

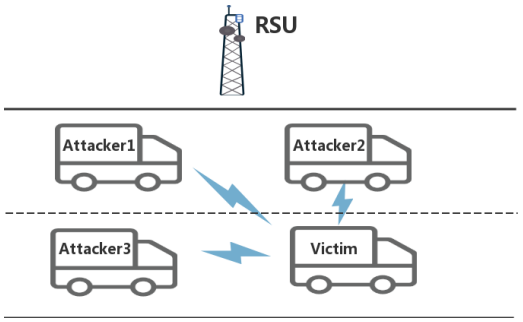
INDEX TERMS Artificial intelligence, distributed denial-of-services, intrusion detection, intelligent transportation systems, spark, vehicular ad hoc networks.

I. INTRODUCTION

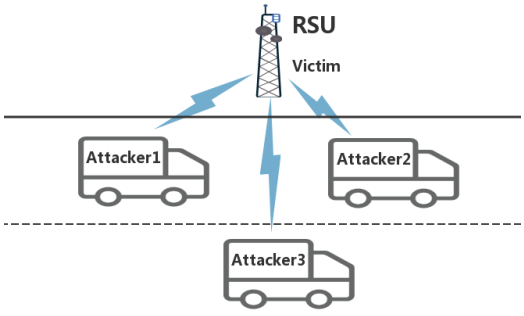
Vehicular Ad hoc NETwork (VANET) is regarded as a unique form of Mobile Ad hoc NETwork (MANET), since the communication nodes are mainly vehicles, and it is a key part of the Intelligent Transportation Systems (ITS) [1]–[3]. Information sharing between nodes in VANET is achieved through two patterns of communication, namely vehicle-to-vehicle (V2V) and vehicles-to-infrastructure (V2I) communication [4], [5]. VANETs deployed by connected vehicles and infrastructure expand security vulnerabilities inherited from wireless communications, particularly in distributed denial-of-service (DDoS) attacks [6]. As shown in Figure 1, DDoS attacks can be launched easily in VANET through two patterns of communication mentioned above. Once a vehicle is under DDoS attack, it can hardly share its driving information to other nodes and access service(e.g. traffic condition reporting and accident warning) provided by the infrastructure which poses a potential threat to safe driving.

The targets of DDoS attacks are various including system resources, network bandwidth and other resources, and its attack mode is changing every year. Thus, a DDoS attack is the main threat to system availability. To provide high availability of VANETs, a scalable, reliable and robust network intrusion detection system should be developed to efficiently mitigate DDoS attacks. However, with the development of cloud computing, Internet of Things (IoT) and Artificial Intel-

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



(a) DDoS attacks launched through V2V communication pattern



(b) DDoS attacks launched through V2I communication pattern

FIGURE 1. DDoS attacks launched in VANET through two patterns of communication.

ligence (AI), attackers can launch large-scale DDoS attacks at low cost. It is more difficult to detect and defend DDoS attacks. And in most cases, DDoS network traffic is similar to normal one [7]. Therefore, how to detect and identify DDoS from a large number of network traffic is a complex and challenging work.

Most common mechanisms to detect and prevent DDoS consist of attack prevention, attack detection, and attack reaction [8]. Network intrusion detection system (NIDS) is becoming a necessary component of network security construction [9]. It detects the abnormal usage of the system by monitoring and analyzing the behavior of a network to detect the attack [10]. There are two types of NIDS, namely, Signature-based NIDS and Anomaly-based NIDS. Signature-based detection recognizes attacks by collecting and determining whether there is a certain pattern or signature of prior attacks in traffic. However, it is not very effective in DDoS attack detection because attackers often change the types and methods of attacks and thence it is difficult to determine the pattern or signature of an attack [11]. In another detection mechanism, namely Anomaly-based detection, normal behavior is learned through a long period of training. When a DDoS attack occurs on a system, the relevant traffic will lead to abnormal network behavior. Thus, NIDS can find the corresponding attacks. In order to overcome the limitations of these two methods, hybrid solutions based on both Signature-based and Anomaly-based techniques have been proposed in the literature [12].

It is still a challenging task for NIDS to detect increasingly complex network attacks. According to [9], [13], AI algorithms, such as Genetic Algorithm (GA) [14], Artificial Neural Network (ANN) [15] and Fuzzy Sets (FS) [16], have been used in the detection of network abnormal attacks. Among them, ANN is one of the representatives. The advantage of using ANN algorithms for unsupervised learning is that they can be relatively effective in detecting DDoS attack packets. However, this kind of methods will not be easily scaled when the data is large and the network structure is complex. It is easy to over-fitting in the training stage. Many traditional machine learning (ML) approaches, e.g., Decision Tree (DT) [17], Support Vector Machine (SVM) [18], Naïve Bayes (NB) [19] and *etc*., have been employed to detect the intrusions. However, their detection performance highly relies on feature engineering. With the development of big data technology, governments, enterprises, and research communities can make use of the vast amount of digital data, and traditional NIDS can no longer meet the data processing requirements of DDoS attack detection. In recent years, many studies focus on detection and defense of DDoS attacks based on a big data framework. Studies [20] combine big data technology with ML technology to satisfy the data processing requirements of DDoS attack detection in the cloud environment. It is a trend of NIDS for DDoS attack detection by using a big data framework to enhance the capacity of processing network traffic and improve detection performance. However, there are many problems with existing researches, including the performance limitation of the detection system, the scalability and stability of the system, and the ability to process large amounts of data.

Aiming to overcome these challenges, we propose a distributed NIDS for DDoS attack Detection based on big data framework in this paper. It uses a distributed architecture to collect and process network traffic and guarantees low latency through a micro-batch data processing model. Apache Spark [21], a memory-based computing framework, is used for feature extraction and model training of cleaned data. To evaluate the performance of the proposed NIDS in DDoS attack detection, two public network traffic datasets, namely, the NSL-KDD [22] and the UNSW-NB15 [23] are used as the benchmarks.

The main contributions of this paper can be summarized as follows:

1) This paper presents a novel big-data-framework based NIDS for DDoS attack detection in VANET. The proposed NIDS consists of two main components: the collection module and detection module. Distributed traffic collection technology is adopted and deployed on some important network nodes to collect traffic efficiently. The ML classification algorithm is accepted to classify the traffic to achieve high accuracy and low FAR.
2) In the collection module, the data processing model of micro-batch is adopted to calculate the collected traffic packet and extract features. At the same time,

the message queue is employed to cache the collected data to ensure the stability of the system.

3) Supervised ensemble ML classifier based on the RF algorithm is applied in the detection module to classify the DDoS attack network traffic and to reduce the false-positive rates. The classified traffic is utilized to update the model so as to improve the classification performance of the model.

The rest of the paper is organized as follows. Section II covers the related work about DDoS attack detection methods. Section III proposes the distributed system architecture of NIDS for DDoS. In Section IV, a series of comparative experiments are performed to estimate the effectiveness and performance of our method. Finally, Section V makes a brief conclusion about this paper and looks ahead of our future work.

II. RELATED WORK

Many detect and response mechanisms for DDoS flooding attacks targeted to service providers have been reported in the last few year [24]. So far, several approaches have been proposed for DDoS attack detection. This section summarizes some of the recent works in DDoS attack detection.

A. MATHEMATICAL METHODS

Chouhan and Peddoju [25] implement the filtering of DDoS attack packets by the hop count of the IP mapping table. For IP attack behavior, the detection accuracy of this method has been up to 90%, and its deployment is convenient. However, this method is vulnerable to distributed attacks. In addition, if the IP2HC update is not timely, legitimate packets may be mistaken for attack traffic and cause false-positives. Moreover, Feinstein *et al.* [26] use the entropy and chi-square test of the network packets to detect DDoS attacks.

B. MACHINE LEARNING METHODS

Wang *et al.* [27] designed a method based on flow mining for abnormal attack detection, demonstrating the worth of this approach on traces from the Slammer and Code Red worms and the MIT Lincoln Laboratories DDoS data. Besides, Huang *et al.* [28] proposed a Deep defense network security architecture, and application of data mining techniques to analyze the alarms collected in the Distributed Intrusion Detection and Prevention System (IDS/IPS). In order to evaluate the effectiveness of the proposed defense architecture, the author uses three different types of data mining tools to realize the prototype and uses it in the DDoS attack detection. It has a good detection effect on attack detection rate and FPR. Gurulakshmi and Nesarani [29] used the SVM algorithm to classify normal and abnormal traffic and predict early anomalous activity. Moreover, Zekri *et al.* [30] proposed a DDoS attack detection system based on C4.5 algorithm. In this system, the signature detection technology is used to generate the decision tree, and the DDoS flood attack is automatically and effectively detected.

C. HYBRID APPROACHES OF BIG DATA AND MACHINE LEARNING

Zhao *et al.* [20] used Hadoop and HBase [31] to deal with a large amount of unstructured data, and designed a neural-network-based detection system for DDoS attacks. Saied *et al.* [15] replaced Hadoop's MapReduce with Apache Spark, and used ANN for attack traffic detection. Apache Spark is a fast and general-purpose cluster computing system that can be used with other components of Hadoop, such as Yarn for resource management, and HDFS for data storage. Recently, Alsirhani *et al.* [32] used multiple classification methods in spark for traffic anomaly detection, and used Fuzzy Logic to select the classification algorithms and verified it on Matlab. The experimental results showed that the tree-based classification algorithms have a better classification effect on traffic classification.

RF algorithm [33] is an ensemble algorithm based on DT. Leo Breiman combined his idea of bagging algorithm [34] with Ho's proposed random subspace algorithm [35] and CART decision tree [36] to put forward the RF algorithm in 2001. It obtains the training subset of each tree through Bagging, and randomly extracts some features as feature subspaces for node splitting and decision tree construction at each node. Finally, all the created trees are integrated, and the final classification is chosen by voting.

In general, the performances of the NIDS for DDoS attack depends on the system architecture, the detection algorithm and the distribution characteristics of the data. Therefore, this paper proposed a novel big-data-framework based NIDS for DDoS attack detection inspired by the previous studies.

III. THE PROPOSED DISTRIBUTED NIDS FOR DDOS ATTACK DETECTION

This section presents a distributed NIDS for DDoS attack detection. Figure 2 depicts the architecture of the whole system. The system consists of two main modules: network traffic collection module and network traffic detection module. Considering the specific scenario of DDoS attack detection in VANET environment, we use big data technology to meet the demand for data processing in terms of a large data volume DDoS attack detection. In the following sections, we introduce the functions and technical principles of each module of the proposed NIDS at first. Then we focus on the data processing and classification algorithms of the detection module.

A. COLLECTION MODULE

The collection module is mainly composed of three parts: Packet collector, Message Queue and Packet Parser. The traffic packet collector is located at the primary network node. It captures traffic passing through the node using the traffic packet collection tool (such as Libpcap). Libpcap [37] is a user-level packet capture interface that provides a convenient application framework for low-level network monitoring. It provides users with a unified and easy-to-use packet capture

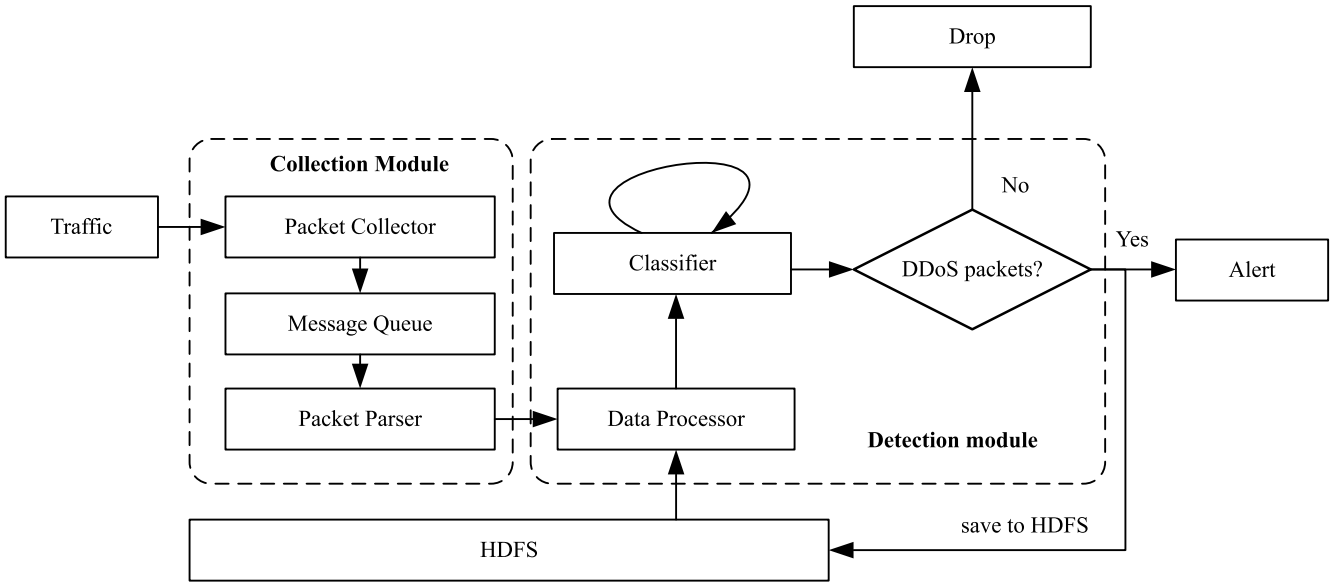


FIGURE 2. The overall architecture of the proposed Distributed DDoS Network Intrusion Detection System.

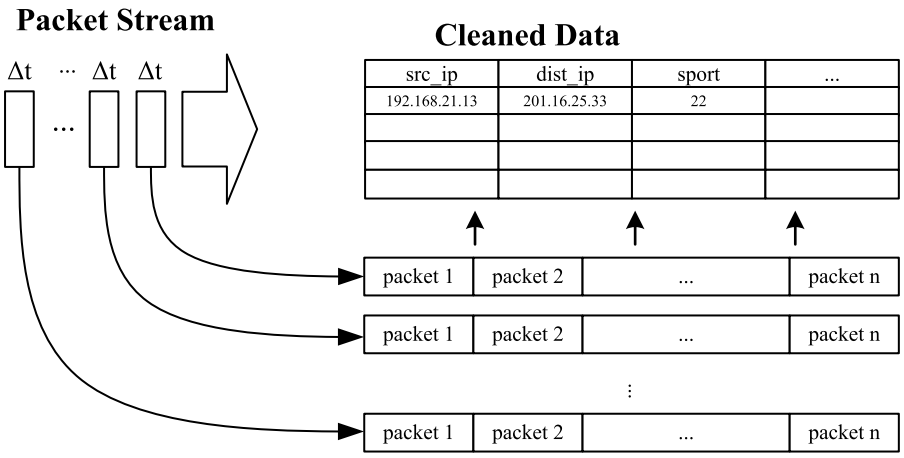


FIGURE 3. Packet stream periodic micro-batch processing.

API. So users can easily develop traffic collection tools based on it.

Due to the large bandwidth of network nodes, if the collected traffic is not processed in a timely manner, the traffic data of this node will continue to accumulate, affecting the stability and availability of the whole system. Therefore, we use the distributed message queue (MQ) [38] to cache the collected network traffic. Message Queuing can well balance the inconsistency of network traffic data collection and processing by decoupling data packet capturing and packet parsing.

The micro-batch data processing model is used in packet parser. First of all, a series of fixed time windows are used to periodically segment the real-time collected traffic packets to generate many small datasets. Secondly, the packets in the same dataset are parsed in batches to extract attributes. The captured Δt time interval packets from the packet stream are treated as a dataset and the attributes extracted from this dataset is appended to the cleaned data (Figure 3). Finally, the cleaned data is used to the detection module. Algorithm 1 shows the process of Packet Parser program. The traffic packets captured and cached are the input of the algorithm. After the initialization process, a tumbling window will be created every Δt time. In each time the window calculation is triggered, and the packets in the time window are taken out from the MQ. Each captured packet is parsed according to the TCP/IP network programming model. Then, according to the protocols (like Ethernet, IP, TCP, UDP, etc.), attributes are extracted from the packet as the basic features of a single traffic packet. At the same time, inspired by the characteristics of the KDD99 dataset [22], the statistical attributes of the traffic in the same time window are calculated. At last,

Algorithm 1 Feature Extraction

Input: The traffic packets in binary format
1: **loop**
2: load packets from MQ with time interval Δt as P
3: **for** $i \leftarrow 1$ *to* n **do** //n is the number of packets
4: Parse packet(i) according to different network protocols (Ethernet, IP, TCP, UDP, etc.)
5: Calculate statistical characteristics
6: **end for**
7: **end loop**
Output: structured packets data

the basic attributes and statistical attributes are sent to the detection module for abnormality detection.

B. DETECTION MODULE

The detection module mainly consists of two parts: Data Processor and Classifier. The Data Processor is applied to further handle the traffic features extracted and calculated from the Collector module. The handling process includes the removal of duplicate samples and data type conversion. The processed data is submitted into the trained DDoS classifier for the detection of network attacks. If it is detected as a DDoS attack, the traffic attribute of the attack will be saved for model update and evidence collection. At the same time, an alarm will be issued to notify the network administrator. The main steps of data preprocessing are as follows:

i: Data containing basic attributes and statistical attributes of packets is loaded. Meanwhile, the dataset of the same batch will be traversed and all properties in each row will be converted to float type.

ii: All the attributes of numerical types are transformed into DataFrame [39], because the detection algorithms use Spark as the calculation engine and DataFrame is the basic data structure of Spark.

iii: VectorAssembler [40] is used to transform the DataFrame into a feature adapting to the classifier. VectorAssembler is a converter that combines a given number of columns into a single vector column. In each row, the values of the input columns are concatenated into a vector in the specified order.

iv: MinMaxScale [41] algorithm is used to generate the final features for detection. It can normalize multiple feature vectors and rescale the value of feature vectors to [0,1]. Since different features are in different orders of magnitude, in order to improve the calculation accuracy, feature vectors need to be normalized. The rescaled value for a feature A is calculated as

$$Rescaled(a_i) = \frac{a_i - A_{min}}{A_{max} - A_{min}} * (max - min) + min \quad (1)$$

where $max = 1$ and $min = 0$

The classifier is a trained RF based on Spark. DDoS attack detection can be regarded as a binary classification problem, where DDoS attack traffic and normal traffic should be classified. When DDoS traffic is detected, we store attack traffic in HDFS for subsequent traceability and model update, as well as notify the network administrators.

C. SPARK-ML RF-BASED ALGORITHM

RF is an ensemble classification algorithm based on DT. It is one of the most widely used classifiers in massive data processing. In RF, the bootstrap sampling method is employed to generate k different subsets of training data, and then k decision trees are constructed by training these subsets. Finally, all trained decision trees are composed of an RF. All decision trees predict each sample of the test data set and return the final classification results by the voting of these trees.

Formally, for a p-dimensional random vector $V = (V_1, \ldots, V_p)^T$ representing the input and a random variable Z representing the response. The goal is to find a prediction function $f(V)$ for predicting Y. The prediction function is determined by a loss function $L(Z, f(V))$ and defined to minimize the expected value of the loss

$$E_{VZ}(L(Z, f(V))) \quad (2)$$

where the E_{VZ} denote expectation with respect to the joint distribution of V and Z.

The reason for choosing RF as DDoS attack detector is based on the following two advantages. On the one hand, the samples and features of decision trees are randomly selected, and the final classification results are derived from multiple decision trees through a voting mechanism. Hence, they have good tolerance to the noises and outliers, and have good generalization ability. On the other hand, multiple decision trees in RF algorithm can be trained in parallel, so they are especially suitable for distributed computing.

Apache Spark is a memory-based distributed computing framework for large data processing. It uses RDD (Resilient Distributed Datasets) [42] as the basic data structure. RDD is an abstraction of data structure in Spark, representing a set of immutable, partitioned storage, and can be computed in parallel. An RDD has multiple partitions. The size of partitions determines the granularity of RDD computation, while the calculation of each RDD partition is performed in a separate task.

The main process of Spark-ML RF-Based algorithm training is shown as follow:

i: The label column and feature column are extracted from the dataset and converted into the RDD

ii: Calculate the number of splits, bins and characteristics sampled by each node in a single decision tree according to the dataset

iii: Find the splits and the corresponding bins of all features

iv: Bootstrap is open up to sample from the original RDD, then build T decision tree training by RDD

v: Create a FIFO node queue named *nodeStack* that stores the root node of each tree

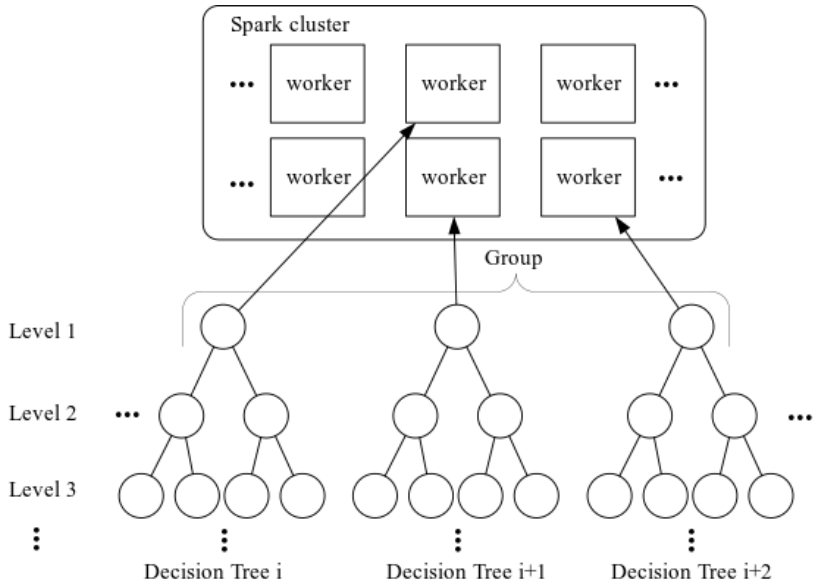


FIGURE 4. Process of the level-wise training and group-wise training.

vi: Traversing the node queue, process every node until the *nodeStack* is empty. The nodes in *nodeStack* is sent to the spark worker to split. When a node completes the split, it will be dequeue. At the same time, the new nodes will enqueue and participate in the next iteration.

vii: After the nodes in the queue are split, the detection model training ends.

Before starting to train, We need to get the labeled data S and its size n. The *ith* example S_i is formalized as (x_i, y_i),where $x_i \in \mathbb{R}^d$ contains d characteristics and $y_i \in \{0, 1\}$ is the class of x_i. Because it is a binary classification, the normal traffic label is defined as 0, whereas the DDoS attack traffic label is denoted as 1. Meanwhile, other hyper-parameters in RF algorithm should also be determined. For example, the number of trees, the maximum growth depth of each tree, the maximum number of feature boxing, the ratio of sample sampling, *etc.*

Since Spark uses the basic data structure of RDD internally, we extract the label column and feature column from the input training data set S to build the RDD of LabelPoint. A LabelPoint encapsulates a sample label and a feature vector. Next, the RDD of LabelPoint is utilized to generate training parameters, namely the number of splits, bins, and characteristics, and calculation logic for each decision tree. After that, the splits and the corresponding bins (interval between the splits) are calculated using a sample of input data. By using the random sampling method, the sample subset for each decision tree training is generated and the number of samples of each sample in the subset of different trees will be cached in memory. On the basis of generated training data and calculated algorithm parameters, the construction of the RF starts.

Level-wise training and group-wise training are jointly used to construct multiple decision trees in Spark. First of all, a FIFO node queue is created. The root nodes of T decision trees are stored in the node queue. The training of RF is to divide these root nodes. In each iteration, according to the memory of each worker in spark cluster, the nodes are distributed to different workers in groups to perform the computation task of node splitting (Figure 4). Multiple decision trees are trained simultaneously in a group. Each time the split child node is appended to the end of the queue, a new layer is formed, waiting for the next iteration. When all tree nodes are no longer split, that is, when the node stack is empty, the RF is finished training.

When the tree node is split, the worker selects the best split by calculating the Gini impurity of the different possible splits. Given a group of examples X, the Gini impurity is formulized as:

$$I(X) = 1 - \sum_{i=1}^{m} p_i^2 \tag{3}$$

where p_i denotes the probability of examples in A that belongs to class i. As the examples X is split into $(X_1, X_2, \ldots, X_m)$ according to the attribute A, the impurity value decreases as:

$$I(X, A) = \sum_{i=1}^{m} \frac{|X_i|}{|X|} I(X_i) \tag{4}$$

The best splitting attribute can be obtained by minimizing the $I(X, A)$. When RFA is used for classification, T decision trees perform classification and voting on the preprocessed data. The category with more votes will be the final classification result.

IV. EXPERIMENT AND DISCUSSION

A. DATASET

In this section, we briefly introduce two datasets about network intrusion detection, including NSL-KDD dataset and UNSW-NB15 dataset, and then experimental settings are also

offered. In order to comprehensively validate the performance of the proposed detection method, several comparisons are finished and the effectiveness of some key parameters is investigated.

1) NSL-KDD DATASET

NSL-KDD [22] dataset is an authoritative benchmark for measuring network intrusion detection methods. It is derived from the KDDCUP99 dataset, which has a salient drawback that a large number of useless duplicate records exist in it. This may cause the detection algorithm focusing on frequent records. Therefore, NSLKDD removes repeated records from the KDDCUP99 to eliminate the classifier's bias. Specifically, NSLKDD contains two training sets ('KDDTrain' and 'KDDTrain_20percent') and two test sets ('KDDTest+' and 'KDDTest-21'), as well as defines 41 features describing the basic content and statistical information of the network. It is divided into five network categories, including, Denial-of-Service (DoS), unauthorized access to the local supervisor privileges (U2R), unauthorized access from a remote machine (R2L), and scanning the network to find known vulnerabilities (Probing). The details of the categories are shown in Table 1. We select the samples labeled Normal and Dos from NSL-KDD as the dataset, as shown in the **boldface** of the table.

TABLE 1. The detail of NSL-KDD dataset.

	Training Dataset		Testing Dataset	
	KDD_Train	KDD_Train_20percent	KDDTest+	KDDTest-21
Normal	**67,343**	13,449	**9711**	**2152**
Probe	11,656	2289	2421	2402
DOS	**45927**	9234	**7458**	**4342**
U2R	52	11	200	200
R2L	995	209	2754	2754
Total	125,973	25,192	22,544	11,850

2) UNSW-NB15 DATASET FOR INTRUSION DETECTION

Since KDDCUP99 and NSL-KDD datasets are old and no longer well representative of the current network environment. Moustafa and Slay [23] proposed a new dataset called UNSW-NB15 using the IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS). The UNSW-NB15 is quite different from KDDCUP99 and NSL-KDD, which reflects a more complex threat environment. This new dataset develops more attack categories and simulates the circumstances with highly unbalanced data, presenting significant challenges for hybrid intelligent algorithms designing for intrusion detection and prevention. Moreover, the UNSW-NB15 includes 49 features, being more abundant than NSL-KDD. About the attack category, nine types of standard network attacks are considered, containing Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms, as shown in Table 2. Similar to NSL-KDD, we select the normal and DoS from UNSW-NB15 dataset, as shown in the **boldface** of the table, where training the classifier with the training set, and then validated it on the test set.

TABLE 2. The detail of UNSW-NB15 dataset.

Type	Training Dataset	Testing Dataset
Normal	**56,000**	**37,000**
Generic	40,000	18,871
Exploits	33,393	11,132
Fuzzers	18,184	6062
DoS	**12,264**	**4089**
Reconnaissance	10,491	3496
Analysis	2000	677
Backdoor	1746	583
Shellcode	1133	378
Worms	130	44
Total	175,341	82,332

B. EXPERIMENTAL SETTINGS

1) EXPERIMENTAL PLATFORM

All experiments were performed on a single server. The CPU of the server is Intel(R) Xeon(R) CPU e5-2603 v3@1.60GHZ. Five virtual machines are set up with KVM, each with 10G of memory and two-core CPU. A cluster of five nodes, including HDFS and YARN, is installed in virtual machines based on Hadoop 2.8.0. Data processing and detection algorithms are implemented using Scala 2.12.6. Then, the program is submitted to the yarn cluster for execution in a yarn-cluster mode.

2) DATA PREPROCESSING

The main purpose of the proposed NIDS is for the detection of DDoS attack traffic, which is identified from the normal traffic. Therefore, we firstly filter the two datasets separately and select the data samples labeled Normal and DoS. Then, features of the nominal in the selected dataset are replaced with the type of numerical. Finally, the feature is normalized to the interval [0, 1] using the data normalization method MinMaxScaler [41].

3) EVALUATION METRICS

In this paper, four standard classification performance measurements are adopted to comprehensively estimate our proposed NIDS. These classification measures are all based on four elements: True positives (TP), Ture negatives(TN), False positives(FP), False negatives(FN). Their definitions are as follows:

- TP: the number of correctly classified DDoS traffics.
- TN: the number of correctly classified normal traffics
- FP: the number of incorrectly classified DDoS traffics
- FN: the number of incorrectly classified normal traffics

The representations of utilized metrics are given as follows.

TABLE 3. Effectiveness comparison of different detectors in NSL-KDD dataset.

Classifiers	KDDTest+				KDDTest-21			
	Accuracy(%)	Recall(%)	FAR(%)	F1-Measure(%)	Accuracy(%)	Recall(%)	FAR(%)	F1-Measure(%)
RF	99.9476	**99.9464**	0.0515	99.9397	**99.8922**	**99.9309**	**0.1859**	**99.9194**
NB	94.6415	99.2765	8.2772	93.4715	86.3412	98.6483	28.5083	88.7622
LR	**99.9592**	**99.9464**	**0.0309**	**99.9531**	99.8614	99.9079	0.2323	99.8964
SVM(RBF)	86.6387	92.8050	16.7011	82.9948	64.8753	85.4978	51.7962	68.5162
SVM(Linear)	89.4403	92.6295	12.5616	87.1227	72.2821	86.3975	44.1372	77.0232
SVM(Sigmoid)	86.6562	91.2239	15.9695	83.3078	64.9215	82.8872	51.8772	69.5455
GBDT	93.1097	98.8327	10.3127	91.4788	81.7832	97.7335	34.7881	84.5380
XGBoost	91.6477	96.5102	11.2971	89.7086	77.9181	93.2738	38.5450	81.3815

Accuracy indicates the percentage of correctly identifying DDoS attacks in the dataset, whose calculation is defined as Equation (5).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

The Recall represents the ratio that the DDoS traffics are correctly classified. The recall of proposed NIDS is measured by Equation (6).

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

The False Alarm Rate (FAR) represents the ratio that the normal traffics are recognized as the DDoS traffics. The FAR can be measured by Equation (7).

$$FAR = \frac{FP}{FP + TN} \tag{7}$$

F1-Measure is a comprehensive evaluation indicator. It is defined as the weighted harmonic average of Precision and Recall, which is defined as Equation (8).

$$F1 - Measure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{8}$$

where the *Precision* is the fraction of correctly classified DDoS traffics to all attack records and is denoted as Equation (9).

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

It is obvious that a well-performed detection model should have high accuracy, high recall, high F1-measure and low FAR.

C. EXPERIMENTAL RESULTS

1) COMPARISON WITH DIFFERENT CLASSIFICATION APPROACHES USED IN NIDS

To validate the effectiveness of the SPARK-ML RF-Based algorithm, we compare it with other classic ML approaches, such as Naïve Bayes [43], Logistic Regression [44], Support Vector Machine (SVM) [45], Gradient Boosting Decision Tree (GBDT) [46] and XGBoost [47]. For the SVM algorithm, different kernel functions are used [48], including RBF, Linear and Sigmoid.

TABLE 4. Effectiveness comparison of different detectors in UNSWSS-NB15 dataset.

Classifiers	Accuracy(%)	Recall(%)	FAR(%)	F1-Measure(%)
RF	**98.7519**	**98.9793**	**1.0752**	**96.4747**
NB	93.4226	94.3418	6.7128	78.6495
LR	82.0345	N/A	N/A	N/A
SVM(RBF)	94.0759	99.5419	6.6802	80.3307
SVM(Linear)	95.5628	98.0539	4.8453	86.1519
SVM(Sigmoid)	93.2981	99.7799	7.5288	77.1101
GBDT	98.5688	98.8065	1.4797	95.8996
XGBoost	98.5922	98.5566	1.4004	95.9796

The results of the NSLKDD dataset are shown in Table 3. It can be found that the RF and LR algorithms outperform other comparing algorithms on both KDDTest+ and KDDTest-21. They obtain the accuracy 99.9476% and 99.9592% on KDDTest+ dataset, and 99.8922% and 99.8614% on KDDTest-21 dataset respectively. However, SVM with different kernel is not performing well. Moreover, it can be seen from Table 5 and Table 6 that the Naïve Bayes has high FAR on two testing datasets because the number of incorrectly classified DDoS is higher than that in other algorithms.

Between two comparing test sets, an apparent observation is that the accuracy, recall and F1-Measure of 'KDDTest+' are higher than those in 'KDDTest-21' respectively, whilst FAR is lower than 'KDDTest-21'. The reason for this is that the 'KDDTest-21' test set contains some unknown DoS attacks in the training set, and the unknown attack type increases the detection difficulty of the model.

The results for the UNSW-NB15 dataset are shown in Table 4. Obviously, RF is significantly better than other classification algorithms, followed by XGBoost, GBDT, SVM(Linear), SVM(RBF), Naïve Bayes SVM(Sigmoid) and LR. Because all records in the test set are detected as Normal in LR, the estimating consequences except Accuracy are ignored. The failure of LR for distinguishing Normal/DoS

TABLE 5. Confusion matrix on KDDTest+.

RF	Predicted class	
Actual class	DDoS	Normal
DDoS	7454	5
Normal	4	9706

NB	Predicted class	
Actual class	DDoS	Normal
DDoS	6586	872
Normal	48	9663

LR	Predicted class	
Actual class	DDoS	Normal
DDoS	7454	3
Normal	4	9708

SVM(RFB)	Predicted class	
Actual class	DDoS	Normal
DDoS	5598	1860
Normal	434	9277

SVM(Linear)	Predicted class	
Actual class	DDoS	Normal
DDoS	6133	1325
Normal	488	9223

SVM(Sigmoid)	Predicted class	
Actual class	DDoS	Normal
DDoS	5717	1741
Normal	550	9161

GBDT	Predicted class	
Actual class	DDoS	Normal
DDoS	6350	1108
Normal	75	9636

XGBoost	Predicted class	
Actual class	DDoS	Normal
DDoS	6250	1208
Normal	226	9485

TABLE 6. Confusion matrix on KDDTest-21.

RF	Predicted class	
Actual class	DDoS	Normal
DDoS	4339	4
Normal	3	2148

NB	Predicted class	
Actual class	DDoS	Normal
DDoS	3503	839
Normal	48	2104

LR	Predicted class	
Actual class	DDoS	Normal
DDoS	4338	5
Normal	4	2147

SVM(RFB)	Predicted class	
Actual class	DDoS	Normal
DDoS	2482	1860
Normal	421	1731

SVM(Linear)	Predicted class	
Actual class	DDoS	Normal
DDoS	3017	1325
Normal	475	1677

SVM(Sigmoid)	Predicted class	
Actual class	DDoS	Normal
DDoS	2601	1741
Normal	537	1615

GBDT	Predicted class	
Actual class	DDoS	Normal
DDoS	3234	1108
Normal	75	2077

XGBoost	Predicted class	
Actual class	DDoS	Normal
DDoS	3134	1208
Normal	226	1926

may be contributed by the similarity between these two network categories (Table 7).

Combining the performance of different classification algorithms on the two datasets, we can find that with respect to the proposed NIDS, the RF algorithm can achieve good results in DDoS attack detection, accompanying with the low FAR.

2) DISCUSSION ABOUT EFFICIENCY OF DIFFERENT PARAMETERS IN RF

In this part, we mainly investigate the effect of the sampling rate, the maximum depth of trees and the number of trees of RF. We use UNSW-NB15 as testing dataset.

TABLE 7. Confusion matrix on UNSW-NB15.

RF	Predicted class	
Actual class	DDoS	Normal
DDoS	8270	3994
Normal	496	55504

NB	Predicted class	
Actual class	DDoS	Normal
DDoS	11658	606
Normal	246	55754

LR	Predicted class	
Actual class	DDoS	Normal
DDoS	0	12264
Normal	0	56000

SVM(RFB)	Predicted class	
Actual class	DDoS	Normal
DDoS	8258	4006
Normal	38	55962

SVM(Linear)	Predicted class	
Actual class	DDoS	Normal
DDoS	9422	2842
Normal	187	55813

SVM(Sigmoid)	Predicted class	
Actual class	DDoS	Normal
DDoS	7706	4558
Normal	17	55983

GBDT	Predicted class	
Actual class	DDoS	Normal
DDoS	11425	839
Normal	138	55862

XGBoost	Predicted class	
Actual class	DDoS	Normal
DDoS	11471	793
Normal	168	55832

In order to argue the influence of the sampling rate in RF, we fix the maximum depth of trees to 20. In the cases where the number of trees is 90, 100 and 110, we compare the accuracy and FAR as the sampling rate varies from 0.2 to 0.95. Through a large number of experiments, we find that the experimental results will become stable when the maximum depth of trees is set to 20 and the number of trees is 90 100, and 110.The accuracy and FAR for each sampling rate is provided in figure 5. It is obvious that the sampling rate has a great influence on detection performance. Overall, with the increase of sampling rate, the accuracy shows an upward trend, while the FAR shows a downward trend. When the sampling rate reaches 0.55, the detection performance tends to be stable. Finally, the accuracy rate was stable at 98.67% and FAR was stable at 1.17%. This experimental result indicates that the low sampling rate leads to the under-fitting of model training.

To explore the impact of the maximum depth of trees in RF, we set the sampling rate to 0.9 according to the previous experiment. The experimental results are shown in Figure 6. It is obvious that with the increase of the maximum depth of trees, the detection performance has been greatly improved. When the maximum depth of trees reaches 16, the accuracy and FAR of the model will be stable gradually.

Furthermore, in order to study the efficiency of the number of trees in RF, we set the maximum depth of trees to 20 based on experiments mentioned before. In the cases of a sampling rate changing from 0.6 to 0.9, we compare the effects of different numbers of trees on detection performance. The accuracy and FAR for each sampling rate are provided in Figure 7. According to Figure 7, a prominent phenomenon is that as the number of trees increases, the

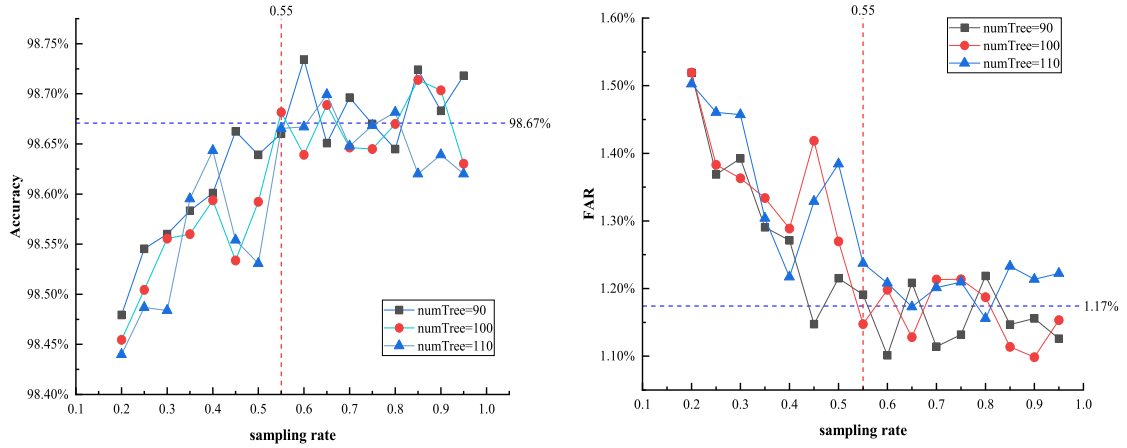


FIGURE 5. The effects of the sampling rate.

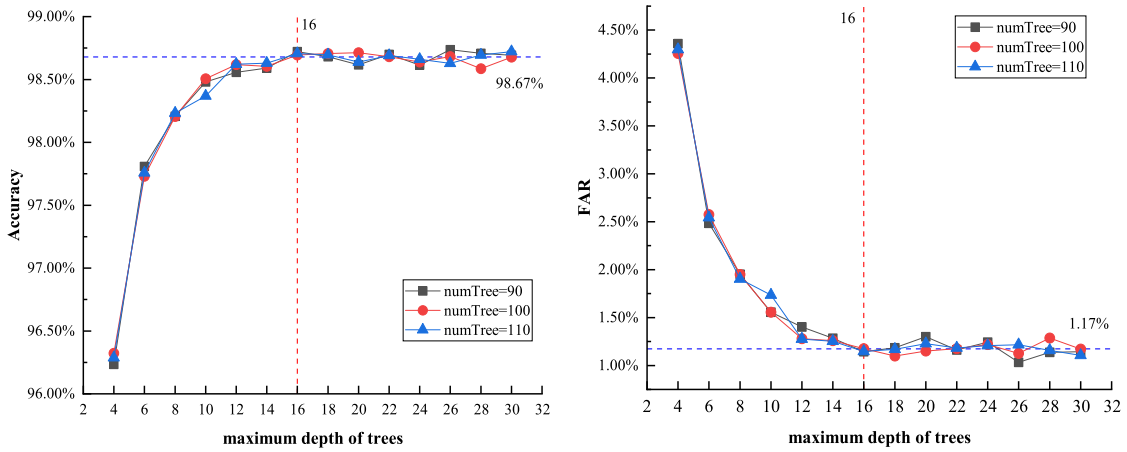


FIGURE 6. The effects of max depth of trees.

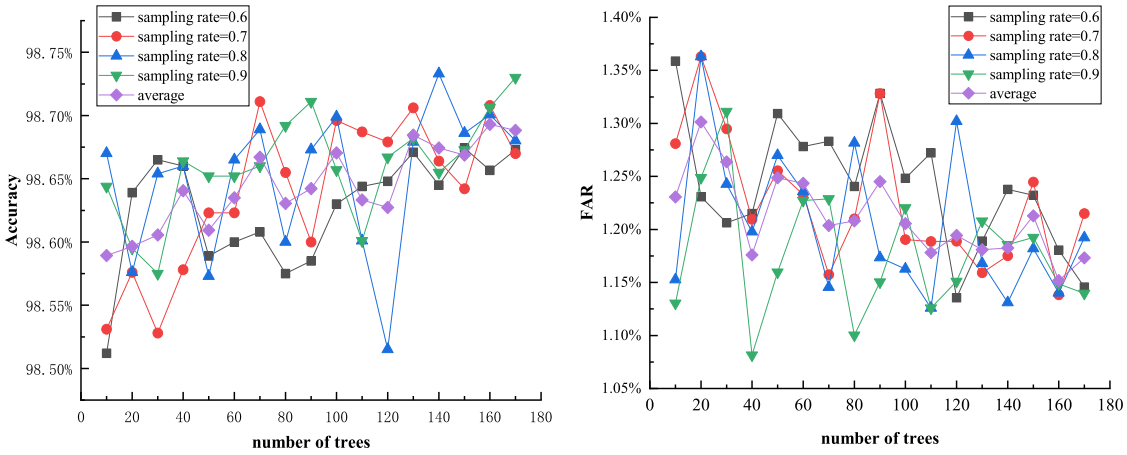


FIGURE 7. The effects of number of trees.

detection performance firstly increases and then stabilizes. These experimental results show that as the number of trees increases, the RF can better fit the training set, thus achieving better results. However, the increase in the number of trees will also result in higher training costs.

3) COMPARISON WITH STATE-OF-THE ART

For the purpose of verifying the obtained results, our method is also compared with several state-of-the-art DDoS attack detection methods. NSL-KDD dataset is used in this part because it is the most commonly used benchmark dataset in

TABLE 8. Comparison of the proposed Spark-ML RF-Based algorithm with the state-of-the-art DDoS attack detection methods.

Approach	Accuracy(%)	FAR(%)
Proposed NIDS	**99.95**	0.05
Mohamed	98.23	0.33
Hybrid-ADE	97.40	0.45
Marliboost	96.38	**0.01**

stare-of-the-arts. Three state-of-the-arts are utilized, including Mohamed Idhammad's method [49], Hybrid-ADE [50], Marliboost [51]. All comparison results are summarized in Table 8.

On the basis of Table 8, the proposed NIDS in this paper achieves the highest accuracy of 99.95% and a low FAR of 0.05%. The accuracy of our NIDS exceeds the other three latest detection methods. In Mohamed's method, it reduces irrelevant normal data from the network traffic for enhancing the DDoS attack detection accuracy by using entropy estimation method. Hybrid-ADE uses density estimation on the hidden layer of an autoencoder to decline the information loss of network traffic, and hence achieves the improvement of accuracy. However, its FAR is the highest among these start-of-the-art approaches. Although Marliboost achieved the lowest FAR of 0.01% by using an ensemble method, it is limited in term of accuracy.

V. CONCLUSION

This paper proposes a novel big-data-framework based NIDS for DDoS attack detection in VANET. It includes real-time network traffic collection modules and detection modules. The network traffic collection module collects and extracts features in the traffic using micro-batch processing model. The detection module uses a Spark-ML RF-Based classification algorithm to detect DDoS attacks on network traffic. In order to evaluate the effect of the proposed DDoS attack detection system, we mainly carried out experiments on the detection performance of the detection module. The experiment uses two common benchmark datasets, NSL-KDD, UNSW-NB15. Compared with other DDoS attack detection methods, the experimental results have good results in accuracy and false positive rate (FPR). Due to the limitations of the experimental environment, this paper only evaluated the proposed system on public datasets. In future work, we plan to deploy the real environment of the proposed NIDS, and use DDoS tools to launch attacks, and then evaluate the performance of the system.

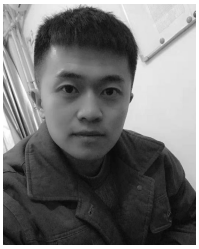
REFERENCES

[1] J. Harri, F. Filali, and C. Bonnet, "Mobility models for vehicular ad hoc networks: A survey and taxonomy," *IEEE Commun. Surveys Tuts.*, vol. 11, no. 4, pp. 19–41, 4th Quart., 2009.
[2] X. Hu, T. H. S. Chu, H. C. B. Chan, and V. C. M. Leung, "Vita: A crowdsensing-oriented mobile cyber-physical system," *IEEE Trans. Emerg. Topics Comput.*, vol. 1, no. 1, pp. 148–165, Jun. 2013.
[3] X. Hu, J. Deng, J. Zhao, W. Hu, E. C.-H. Ngai, R. Wang, J. Shen, M. Liang, X. Li, V. C. M. Leung, and Y.-K. Kwok, "SAfeDJ: A crowd-cloud codesign approach to situation-aware music delivery for drivers," *ACM Trans. Multimedia Comput., Commun. Appl.*, vol. 12, no. 1s, p. 21, 2015. doi: 10.1145/2808201.
[4] X. Wang, Z. Ning, M. C. Zhou, X. Hu, L. Wang, Y. Zhang, F. R. Yu, and B. Hu, "Privacy-preserving content dissemination for vehicular social networks: Challenges and solutions," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1314–1345, 2nd Quart., 2018.
[5] X. Hu, J. Zhao, B. C. Seet, V. C. M. Leung, T. H. S. Chu, and H. Chan, "S-Aframe: Agent-based multilayer framework with context-aware semantic service for vehicular social networks," *IEEE Trans. Emerg. Topics Comput.*, vol. 3, no. 1, pp. 44–63, Mar. 2015.
[6] S. Parkinson, P. Ward, K. Wilson, and J. Miller, "Cyber threats facing autonomous and connected vehicles: Future challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 2898–2915, Nov. 2017.
[7] B. Zhang, T. Zhang, and Z. Yu, "DDoS detection and prevention based on artificial intelligence techniques," in *Proc. 3rd IEEE Int. Conf. Comput. Commun. (ICCC)*, Dec. 2017, pp. 1276–1280.
[8] X. Yuan, C. Li, and X. Li, "DeepDefense: Identifying DDoS attack via deep learning," in *Proc. IEEE Int. Conf. Smart Comput.*, May 2017, pp. 1–8.
[9] S. X. Wu and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review," *Appl. Soft Comput.*, vol. 10, no. 1, pp. 1–35, 2010.
[10] G. B. White, E. A. Fisch, and U. W. Pooch, "Cooperating security managers: A peer-based intrusion detection system," *IEEE Netw.*, vol. 10, no. 1, pp. 20–23, Jan. 1996.
[11] K. Kalkan, G. Gur, and F. Alagoz, "Filtering-based defense mechanisms against DDoS attacks: A survey," *IEEE Syst. J.*, vol. PP, no. 99, pp. 1–13, 2016.
[12] O. Osanaiye, K.-K. R. Choo, and M. Dlodlo, "Distributed denial of service (DDoS) resilience in cloud: Review and conceptual cloud DDoS mitigation framework," *J. Netw. Comput. Appl.*, vol. 67, pp. 147–165, May 2016.
[13] Y. Gao, Y. Liu, Y. Jin, J. Chen, and H. Wu, "A novel semi-supervised learning approach for network intrusion detection on cloud-based robotic system," *IEEE Access*, vol. 6, pp. 50927–50938, 2018.
[14] M. Mizukoshi and M. Munetomo, "Distributed denial of services attack protection system with genetic algorithms on Hadoop cluster computing framework," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, May 2015, pp. 1575–1580.
[15] A. Saied, R. E. Overill, and T. Radzik, "Detection of known and unknown DDoS attacks using artificial neural networks," *Neurocomputing*, vol. 172, pp. 385–393, Jan. 2016.
[16] S. Shamshirband, A. Amini, N. B. Anuar, M. L. M. Kiah, Y. W. Teh, and S. Furnell, "D-FICCA: A density-based fuzzy imperialist competitive clustering algorithm for intrusion detection in wireless sensor networks," *Measurement*, vol. 55, pp. 212–226, Sep. 2014.
[17] S. Sahu and B. M. Mehtre, "Network intrusion detection system using J48 decision tree," in *Proc. Int. Conf. Adv. Comput., Commun. Inform. (ICACCI)*, Aug. 2015, pp. 2023–2026.
[18] Y. Li, J. Xia, S. Zhang, J. Yan, X. Ai, and K. Dai, "An efficient intrusion detection system based on support vector machines and gradually feature removal method," *Expert Syst. Appl.*, vol. 39, no. 1, pp. 424–430, 2012.
[19] R. F. Fouladi, C. E. Kayatas, and E. Anarim, "Frequency based DDoS attack detection approach using naive bayes classification," in *Proc. 39th Int. Conf. Telecommun. Signal Process.*, Jun. 2016, pp. 104–107.
[20] T. Zhao, D. C.-T. Lo, and K. Qian, "A neural-network based DDoS detection system using Hadoop and HBase," in *Proc. IEEE Int. Conf. High Perform. Comput. Commun.*, Aug. 2015, pp. 1326–1331.
[21] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *Proc. USENIX Conf. Hot Topics Cloud Comput.*, 2010, p. 95.
[22] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Int. Conf. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6.
[23] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf.*, Nov. 2015, pp. 1–6.
[24] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 4, pp. 2046–2069, 4th Quart., 2013.
[25] V. Chouhan and S. K. Peddoju, "Packet monitoring approach to prevent DDoS attack in cloud computing," *Int. J. Comput. Sci. Electr. Eng. (IJCSEE)*, vol. 1, no. 2, pp. 2315–4209, 2013.

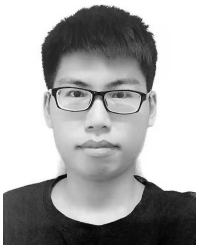
[26] L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred, "Statistical approaches to DDoS attack detection and response," in *Proc. DARPA Inf. Survivability Conf. Expo.*, vol. 1, Apr. 2003, pp. 303–314.
[27] J. Wang, D. J. Miller, and G. Kesidis, "Efficient mining of the multidimensional traffic cluster hierarchy for digesting, visualization, and anomaly identification," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 10, pp. 1929–1941, Oct. 2006.
[28] N.-F. Huang, C.-N. Kao, H.-W. Hun, G.-Y. Jai, and C.-L. Lin, "Apply data mining to defense-in-depth network security system," in *Proc. 19th Int. Conf. Adv. Inf. Netw. Appl.*, Mar. 2005, pp. 159–162.
[29] K. Gurulakshmi and A. Nesarani, "Analysis of IoT Bots against DDoS attack using machine learning algorithm," in *Proc. 2nd Int. Conf. Trends Electron. Inform. (ICOEI)*, May 2018, pp. 1052–1057.
[30] M. Zekri, S. El Kafhali, N. Aboutabit, and Y. Saadi, "DDoS attack detection using machine learning techniques in cloud computing environments," in *Proc. 3rd Int. Conf. Cloud Comput. Technol. Appl. (CloudTech)*, Oct. 2017, pp. 1–7.
[31] M. N. Vora, "Hadoop-HBase for large-scale data," in *Proc. Int. Conf. Comput. Sci. Netw. Technol.*, Dec. 2011, pp. 601–605.
[32] A. Alsirhani, S. Sampalli, and P. Bodorik, "DDoS attack detection system: Utilizing classification algorithms with Apache spark," in *Proc. 9th IFIP Int. Conf. New Technol. Mobility Secur. (NTMS)*, Feb. 2018, pp. 1–7.
[33] A. Cutler, D. R. Cutler, and J. R. Stevens, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 157–176, 2011.
[34] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.
[35] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, Aug. 1998.
[36] L. Breiman, J. Friedman, R. Olshen, and C. J. Stone, "Classification and regression trees," in *The Wadsworth Statistics/Probability Series*, vol. 358. Wadsworth International Group, 1984.
[37] V. Duarte and N. Farruca, "Using libpcap for monitoring distributed applications," in *Proc. Int. Conf. High Perform. Comput. Simulation*, Jun./Jul. 2010, pp. 92–97.
[38] I. Sadooghi, K. Wang, D. Patel, D. Zhao, T. Li, S. Srivastava, and I. Raicu, "FaBRiQ: Leveraging distributed hash tables towards distributed publish-subscribe message queues," in *Proc. IEEE/ACM 2nd Int. Symp. Big Data Comput.*, Dec. 2015, pp. 11–20.
[39] A. Spark. (2018). *Spark SQL, Dataframes and Datasets Guide*. [Online]. Available: http://spark.apache.org/docs/latest/sql-programming-guide.html
[40] A. Spark. (2018). *Vectorassembler*. [Online]. Available: http://spark.apache.org/docs/latest/ml-features.html#vectorassembler
[41] A. Spark. (2018). *Minmaxscaler*. [Online]. Available: http://spark.apache.org/docs/latest/ml-features.html#minmaxscaler
[42] A. Spark. (2018). *Resilient Distributed Datasets (RDDS)*. [Online]. Available: http://spark.apache.org/docs/latest/rdd-programming-guide.html#resilient-distributed-datasets-rdds
[43] G. H. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," in *Proc. 11th Conf. Uncertainty Artif. Intell.*, 2013, pp. 338–345.
[44] A. Cucchiara, "Applied logistic regression," *Technometrics*, vol. 34, no. 3, pp. 358–359, 2012.
[45] S.-X. Lu and X.-Z. Wang, "A comparison among four SVM classification methods: LSVM, NLSVM, SSVM and NSVM," in *Proc. Int. Conf. Mach. Learn. Cybern.*, Aug. 2004, pp. 4277–4282.
[46] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.
[47] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 785–794.
[48] K.-R. Müller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 181–201, Mar. 2001.
[49] M. Idhammad, K. Afdel, and M. Belouch, "Semi-supervised machine learning approach for DDoS detection," *Appl. Intell.*, vol. 48, no. 10, pp. 3193–3208, 2018.
[50] V. L. Cao, M. Nicolau, and J. McDermott, "A hybrid autoencoder and density estimation model for anomaly detection," in *Proc. Int. Conf. Parallel Problem Solving Nature*, 2016, pp. 717–726.
[51] A. S. Boroujerdi and S. Ayat, "A robust ensemble of neuro-fuzzy classifiers for DDoS attack detection," in *Proc. 3rd Int. Conf. Comput. Sci. Netw. Technol.*, 2014.

YING GAO received the bachelor's and master's degree from Central South University of China, in 1997 and 2000, respectively, and the Ph.D. degree from the South China University of Technology, China, in 2006. She is currently a Professor with the School of Computer Science and Engineering, South China University of Technology. She has published more than 30 articles in international journals and conferences. Her current research interests include service-oriented computing technology, software architecture, and network security.

HONGRUI WU received the B.S. degree from the South China University of Technology, China, in 2018, where he is currently pursuing the M.S. degree in computer science and engineering. His research interests include machine learning, pattern recognition, and network security.

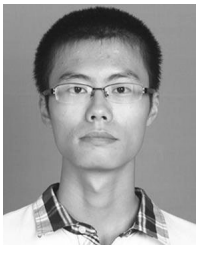


BINJIE SONG is currently pursuing the master's degree in computer science and engineering with the South China University of Technology. His main research interest includes cyberspace security. The main research content is network anomaly behavior recognition based on artificial intelligence. His interests include in evolutionary computing, machine learning, and deep learning.



YAQIA JIN received the B.S. degree in computer science and engineering from the South China University of Technology, in 2017, where he is currently pursuing the M.S. degree in computer science and engineering. His major research interests include cloud computing and network security.

XIONGWEN LUO received the bachelor's degree in computer science from South China Agricultural University, Guangzhou, China, in 2016. He is currently pursuing the MA.Eng. degree with the South China University of Technology, China. His research interests include evolutionary algorithm, dynamic optimization problem, and machine learning, including particle swarm optimization, differential evolution, genetic algorithm, convolutional neural network, and recurrent neural network.



XING ZENG received the bachelor's degree in computer science and technology from the South China University of Technology, in 2018, where he is currently pursuing the master's degree in computer engineering. His major research interests include machine learning and microservice architecture.