Journal of Big Data

RESEARCH Open Access

Apply machine learning techniques to detect malicious network traffic in cloud computing

Amirah Alshammari[1*] and Abdulaziz Aldribi[2]

*Correspondence: afmalshammari@ju.edu.sa
[1] Department of Computer Science, College of Computer, Jouf University, Al Jouf, Saudi Arabia
Full list of author information is available at the end of the article

Abstract
Computer networks target several kinds of attacks every hour and day; they evolved to make significant risks. They pass new attacks and trends; these attacks target every open port available on the network. Several tools are designed for this purpose, such as mapping networks and vulnerabilities scanning. Recently, machine learning (ML) is a widespread technique offered to feed the Intrusion Detection System (IDS) to detect malicious network traffic. The core of ML models' detection efficiency relies on the dataset's quality to train the model. This research proposes a detection framework with an ML model for feeding IDS to detect network traffic anomalies. This detection model uses a dataset constructed from malicious and normal traffic. This research's significant challenges are the extracted features used to train the ML model about various attacks to distinguish whether it is an anomaly or regular traffic. The dataset ISOT-CID network traffic part uses for the training ML model. We added some significant column features, and we approved that feature supports the ML model in the training phase. The ISOT-CID dataset traffic part contains two types of features, the first extracted from network traffic flow, and the others computed in specific interval time. We also presented a novel column feature added to the dataset and approved that it increases the detection quality. This feature is depending on the rambling packet payload length in the traffic flow. Our presented results and experiment produced by this research are significant and encourage other researchers and us to expand the work as future work.

Keywords: IDS, Network traffic, Feature extraction, Dataset, Machine learning

Introduction

The last two years have seen some of the most shared and stark cybersecurity attacks regularly recorded toward networks in different industries. Security specialists expect another record-breaking year of network breaches and data security risks; companies must make themselves aware of the latest threats in circulation to ensure their security countermeasures are up to par. Ninth attacks type are the most significant frequency in the security First report in Garg et al. [9]

In network attacks, the attacker must know active addresses, network topology, and available services. Network scanners can identify open ports on a system, whether TCP or UDP ports, where shared services are related to specific ports, and an attacker could send packets to every port [6]. TCP fingerprinting abilities of how systems react to unauthorized packet formats different vendors TCP/IP stacks answer differently to unauthorized packets. So, the attacker can determine OS by sending numerous combinations of illegal packet options, initiating a connection with an RST packet, or combining other odd and illegal TCP code bits. The attacker could know if a machine is running, whether Linux, Windows, or any other operating system. This information helps to refine the attack and search for weaknesses in specific services and systems to access [25]. For example, DoS attack, the attacker remains network buffers or memory resources in over-busy. They send massive traffic to a system on the network overcoming its capability to respond to legitimate users. The attacker does this by flood systems with ICMP and UDP packets. The most popular packet flood attack takes benefit of the weakness in TCP's three-way handshake. Exhaust a server's ability by leaving half-open connections, so it consumes bandwidth; an attacker would require a more significant relationship than the victim to cut all service [5]. The network must protected from such attacks; robust IDS should deploy before network routers from the company side.

Recently, ML techniques were used to train IDS to capture malicious network traffic. The main idea of IDS based on ML analysis is finding patterns and building an IDS based on the dataset. The IDS can detect adequately. We need to have a real network traffic dataset and proper feature selection to learned enough. Therefore, we aim to propose a detection framework with an ML model to detect malicious traffic rely on a dataset consisting of network traffic attributes to feed IDS, as illustrated in Fig. 1. The dataset called ISOT-CID was created by Aldribi et al. [2] and described in detail in the methodology. The presented model is prepared, constructed, fitted, and evaluated by python language using Sklearn, Numpy, Matplotlib, and Pandas. Our attractive model should construct and fit in memory, so it listens to the extracted feature from network traffic to predict anomalies in real-time. The contribution of our study consists of five things:

1. Extracting network features (Calculated): T-IN, T-OUT, APL, PV, TBP, and novel Rambling can help IDS better detect. These six features added to the dataset are significant to produce a qualitative dataset applicable to the train machine learning model for anomaly detection.
2. Propose a lightweight ML model so it can feed IDS in real-time.
3. Evaluating how calculated features would provide the best classification accuracy using the cross-validation method and split validation.
4. Our model is applicable to be placed on a local network or before the internet router from the company side.
5. Detect whether anomaly or normal traffic.

The remainder of this paper organizes as follows. "Related work" section presents related works, and similar studies are listed. "Detection framework (Our Approach)" section illustrates our framework as a complete solution for detection anomaly, including the machine learning model trained by dataset constructed from network row traffic data. The methodology and experimental results are illustrated in

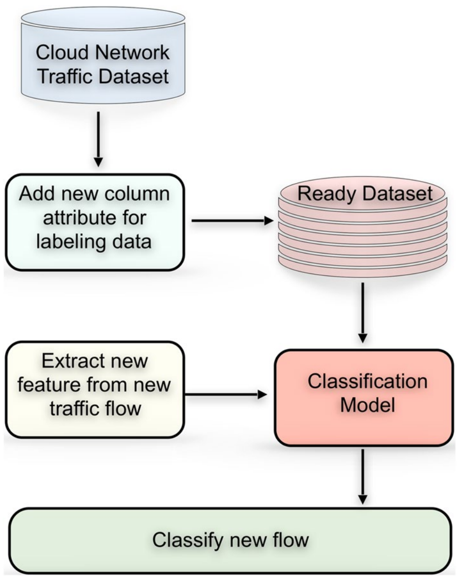
Cloud Network Traffic Dataset
Add new column attribute for labeling data
Ready Dataset
Extract new feature from new traffic flow
Classification Model
Classify new flow

Fig. 1 Detection framework

"Methods" and "Results and analysis" sections, respectively. Finally, the discussion and the conclusion are presented in "Discussion" and "Conclusions and future work" sections, respectively.

Related work

As anomaly detection is most inserting as a researcher issue, there are many explorations and examination efforts in this field. Briefly, we write about significant of them as related works categorized about the kind of proposed solution.

Supervised learning

Parul and Gurjwar [23] used the Decision Tree algorithms classifier to train the IDS in a layered approach. The result of this approach gave a good result in a layered approach used for each layer. They used the Random forest algorithm and gave good results for every layer but have limited U2R attach, which presents a very low-rate classification. The author argues to modify the random forest to improve the result of the U2R layer. The proposed system used the KDDcup99 dataset, which has significant enhancement on the new release of the dataset call NSL_KDD.

Peng et al. [24] presented an IDS based on the decision tree classifier algorithm. The authors compared the result of the work by multi-methods were not only 10% of the dataset; the entire dataset was tested. The experiment results showed that the proposed IDS system was effective. However, when comparing the detection time for each method, the decision tree's time was not the best in the case of guaranteed accuracy. The authors argue that the proposed IDS system can be used in fog computing environments over big data. The proposed system was not tested as a real-time application. The system also used the old version KDD cup 99, a new, recent version with significant development.

The presented paper for Anton et al. [3] shown that some ML anomaly detection algorithms such as SVM and Random Forest achieved well in detecting network traffic anomalies in business networks, where both of them are classifier techniques. The dataset needed for training these models delivered by simulators [14]. The trouble lies in producing sound, actual data that matches the business environment where an anomaly detection model can be applied. There are many opportunities for the allowance of the proposed methods. Data from various resources can be collected, composed, and utilized to increase performance. The overview of context information into the anomaly detection process was capable and encouraged the increase of accuracy.

Additionally, the engagement of trickery technologies as devices for anomaly detection could improve the vision of anomaly behavior. One of the essential dominant requirements is capturing data by attacks exact to business applications in general. The analysis achieved in this work only employs network-based features which, in the same form, residence in home and office devices. The only main diversion was the timing pattern that is strongly interrelated to attacks.

Manna and Alkasassbeh [15] presented a recent approach that used ML, such as decision tree J48, random forest, and REP tree. The proposed technique used SNMP-MIB data for the trained IDS system to detect DOS attack anomalies that may affect the network. The classifiers and attributes were applied to the IP group. The results showed that applying the REP tree algorithm classifier donated the highest performance to all IP set times. The average performance of these three classifiers was accurate enough to be an IDS System. However, it has a limitation that the dataset is extensive and needs more challenges to be used in real-time.

Unsupervised learning

Jianliang et al. [11] proposed applying the K-means clustering algorithm used as ML in intrusion detection. K-means was used for intrusion detection to detect anomalies traffic and divide ample data space efficiently, but it has many drawbacks in cluster dependence. They constructed the intrusion detection model using the k-Medoid clustering algorithm with positive modifications. The algorithm stated selecting initial K-Medoid and verified it to be better than K-means for intrusion detection of an anomaly. The proposed approach has exciting advantages over the existing algorithm, which mostly overwhelms the drawbacks of dependency on primary centroids, dependency on the number of clusters, and unrelated clusters. The proposed algorithm is needed to investigate the detection rate for the root attack and real-time environment.

Qiu et al. [26] presented GAD as a group anomaly detection scheme to pinpoint the subgroup of samples and a subgroup of features that together identify an anomalous cluster. The system was applied in network intrusion detection to detect Botnet and peer-to-peer flow clusters. The approach intended to capture and exploit statistical dependencies that might remain among the measured features. The experiments of the model on real-world network traffic data showed the advantage of the proposed system.

A novel Network Data Mining approach was proposed by Kumari et al. [12]. Their approach uses the K-means clustering technique to feature datasets that are extracted from flow instances. Training data divided into clusters of periods of anomalies and regular flow. While the data mining process was moderately complex, the resulting centroids of clusters are used to detect anomalies in new live observing data with a small number of distance calculations. This approach allows arranging the detection method for accessible real-time detection as part of the IDS system. Applying the clustering technique separately for different services identified by their transport protocol and port number enhances detection accuracy. The presented approach conducted an experiment using generated and actual flow. As the author said, this approach needs several improvements, such as comparing clustering results with different K to determine the optimal number of clusters, considering other features such as the average flow duration, and considering different distance metrics.

Nikiforov [20] used a Cluster-based technique to detect anomalies for Virtual Machines within both production and testing LAB environments with reasonable confidence. Some improvements need to be made to have even welled results in testing environments. This model does not consider the time of day and day of week dependability of the VM load. For example, the night is usually a busy time since many auto-tests were running during the night in the testing infrastructure. Some tests were being run at the same time every day. Based on this, the following improvements in the model might be made. Analyze a detected outlier based on the same time as it was detected but for several days before. Check if this is a case when a load is scheduled and planned. Divide the metrics used for analysis into business days vs. weekends since the load might differ.

Cloud-based techniques

Mobilio et al. [17] presented Cloud-based anomaly detection as a service that used the as-a-service paradigm exploited in cloud systems to announce the anomaly detection logic's control. They also proposed early results with lightweight detectors displaying a promising solution to better control anomaly detection logic. They also discussed how to apply the as-a-service paradigm to the anomaly detection logic and achieving anomaly detection as-a-service. They also proposed an architecture that supports the as-a-service paradigm and can work jointly with any observing system that stores data in time-series databases. The early experimentation of as-a-service with the Clearwater cloud system obtained results demonstrating how the as-a-service paradigm can effectively handle the anomaly detection logic. This approach is fascinating, which integrates new technology of as-a-service in anomaly detection in real-time.

Moustafa et al. [18] proposed a Collaborative Anomaly Detection Framework named CADF for handling big data in cloud computing systems. They provided the technical functions and the way of deployment of this framework for these environments. The proposed approach comprises three modules: capturing and logging network data, preprocessing these data, and a new Decision Engine using a Gaussian Mixture Model [10] and lower–upper Interquartile Range threshold [16] for detecting attacks. The UNSW-NB15 dataset was used for evaluating the new Decision Engine to assess its reliability while deploying the model in real cloud computing systems, and it compared with three ADS techniques. The architecture for deploying this mode as Software as a Service (SaaS) was produced to be installed easily in cloud computing systems.

An ensemble-based multi-filter feature selection method is proposed by Osanaiye et al. [22]. This method achieves an optimum selection by integrating the output of four filter methods. The proposed approach is deployed in cloud computing and used for detecting DDOS attacks. An extensive experimental evaluation of the proposed method was accomplished using the intrusion detection benchmark dataset, NSL-KDD, and decision tree classifier. The obtained result shows that the proposed method decreases the number of features to 13 instead of 41 efficiently. Besides, it has a high detection rate and classification accuracy when compared to other classification techniques.

Barbhuiya et al. [4] presented Real-time ADS named RADS.RADS addresses detecting the anomaly using a single-class classification model and a window-based time series analysis. They evaluated the performance of RADS by running lab-based and real-world experiments. The lab-based experiments were performed in an OpenStack-based Cloud data center, which hosts two representatives, Cloud Applications Graph Analytics and Media Streaming, collected from the CloudSuite workload collection. In contrast, the real-world experiments carried out on the real-world workload traces collected from a Cloud data center named Bitbrains. The evaluation results demonstrated that RADS could achieve 90–95% accuracy with a low false-positive rate of 0–3% while detecting DDoS and crypto-mining attacks in real-time. The result showed that RADS experiences fewer false positives while using the proposed window-based time series analysis than entropy-based analysis. They evaluated the performance of RADS in conducting the training and the testing in real-time in a lab-based Cloud data center while hosting varying 2 to 10 of VMs. The evaluation results suggest that RADS can be used as a lightweight tool to consume minimal hosting node CPU and processing time in a Cloud data center.

Zhang [28] presented Multi-view learning techniques for detecting the cloud computing platform's anomaly by implementing the extensible ML model. They worked on a gap formulated as the pair classification in real- time, which is trained by improving the ELM model's multiple features.

The presented technique automatically fuses multiple features from different sub-systems and attains the improved classification solution by reducing the training mistakes. Sum ranked anomalies are identified by the relation between samples and the classification boundary, and weighting samples ranked retrain the classification model. The proposed model deals with different challenges in detecting an anomaly, such as imbalance spreading, high dimensional features, and others, efficiently via Multi-view learning and feed regulating.

Deep learning techniques

Fernandez and Xu [8] presented a case study using a Deep learning network to detect anomalies. The author said that he achieved excellent results in supervised network intrusion detection. They also showed that using only the first three octets of IP addresses can be efficient in handling the use of dynamic IP addresses, representing the strangeness of DNN in the attendance of DHCP. This approach showed that autoencoders could be used to detect anomalies wherever they trained on expected flows.

Kwon [13] proposed Recurrent Neural Network RNN and Deep Neural Network DNN with ML techniques related to anomaly detection in the network. They also conducted local experiments showing the feasibility of the DNN approach to network flow traffic analysis. This survey also investigated DNN models' effectiveness in network flow traffic analysis by introducing the conducting experiments with their FCN model. This approach shows encouraging results with enhancement accuracy to detect anomalies compared to the conventional techniques of ML such as SVM, random forest, and Ad boosting.

Garg et al. [9] presented a hybrid data processing model for detection anomaly in the network that influences Grey Wolf optimization and Convolution Neural Network CNN. Improvements in the GWO and CNN training approaches improved with exploration and initial population capture capabilities and restored failure functionality. These extended alternatives are mentioned as Improved-GWO and Improved CNN. The proposed model runs in two stages for detection anomaly in the network. At the first stage, improved GWO was utilized for feature selection to attain an ideal trade-off among two objectives to reduce the frailer rate and minimize the feature set. In the second stage, improved CNN was utilized for the classification of network anomalies. The author said that the proposed model's efficiency is evaluated with a benchmark (DARPA'98 and KDD'99) and artificial datasets. They showed the results obtained, which validate that the proposed cloud-based anomaly detection model was superior to the other related works utilized for anomaly detection in the network, accuracy, detection rate, false-positive rate, and F-score. The proposed model shows an overall enhancement of 8.25%, 4.08%, 3.62% in detection rate, false positives, and accuracy, respectively, related to standard GWO with CNN.

Feature extraction

Umer et al. [27] Proposed a flow-based IDS which gets IPFIX/Net Flow records treated as input. Each flows record can have several attributes. Some of these attributes are tacked to the classification model for the decision, while others are used in computational. The significant attributes such as originating IP address destination port play an essential part in the detection judgment's proposed approach. They conducted feature selection to select related attributes required for increasing the performance of the decision. They conducted a preparing process for flow records to convert them into a specific format to be acceptable to anomaly detection algorithms.

Nisioti et al. [21] presented a survey of the unsupervised model for the IDS system. This model's features are extracted from different evidence sources as network traffic, logs from different devices and host machines, etc. Unsupervised techniques proposed to consider as more flexible to the additional features extracted from different sources evidence and do not need regular training back. They also proposed and compared feature selection methods for IDS. This survey finds and uses the optimum feature subset for each class to decrease the computational complexity and time.

Münz et al. [19] presented a detection model for the anomaly in network traffic using a clustering algorithm, which is K-Means for input. The proposed detection model takes captured hypervisor packets and composes them into a stream of packet flows related to operating system time. The model consists of two phases of feature extraction based on the packet's header as a primary feature vector computed for each unique packet. The second phase extracts a separated feature vector to every packet flow related to the primary feature vectors attendant with the packets included in the flow.

Aldribi et al. [2] introduced a hypervisor-based cloud for IDS that includes a novel feature extraction approach depending on the activities of user instances and their related behaviors into the hypervisor. The proposed model intended to detect anomalous behavior into the cloud by tracing statistical variations using a grouping of the gradient descent algorithms and E-Div. The new dataset was introduced as an intrusion detection dataset gathered in a cloud environment available and publicly for researchers. The dataset involves multistage attack scenarios that permit developing and evaluate threat environments relying on cloud computing. They conducted an experimental evaluation using the Riemann rolling feature extraction scheme and produce promising results. The dataset carried the number of communications over encrypted channels, for instance, using protocols like SSH.

Detection framework (our approach)

As shown in Fig. 1, the network traffic dataset consists of flow network traffic attributes described in Aldribi et al. [2] with no label. The proposed dataset extracted from network traffic in different period and contains frame time, source MAC, destination MAC, source IP, source port, destination IP, source port, IP length, IP header length, TCP header length, frame length, offset, TCP segment, TCP acknowledgment, in frequency number, and out frequency number. These attributes of network flow can specify packets, whether anomaly or normal. The formulas shown in Fig. 2 can calculate the in-frequency number, and, similarly, the out-frequency number. Other features that are vital and added to the ISOT-CID dataset are. APL is the average payload packet length for a time interval, PV is the variance of payload packet length for a time interval, and TBP means the average time between packets in the time interval [29].

The main significant thing in our research that we added the novel feature. We believe this novel feature gives support for the ML model in the training process. This feature is called rambling.

Most machine learning models are learning from the diversions of instance values. The closer values can support the classification process more accurately. Depending on our knowledge network flow traffic have many different packet sizes through the various type of contents. The network protocols have limited packet size related to industrial Corporations such as Xerox Ethernet V2, intel, etc. Most of them ranged from (64 to 1518) bytes. Suppose we capture a group of packets that have the same destination IP address in a time interval. Let payload of the packet in specific time T is Vi and Xi is the mean of these V (0,1, 2, ... n) the rambling feature (R) calculate for each instance flow for the interval (t, dt) as the following.

$$R = |Xi - Vi| in(t, dt) \tag{1}$$

This new feature (Rambling feature) can reduce each flow packet size difference, supporting the machine learning algorithm's classification process.

The dataset is labeled related to specific normal IPs, including in the data instances, and used in the ML classification model. The classification model, as presented in Fig. 1 which is trained by an updated ISOT-CID dataset able to classify the new feature extracted from the network data flow, whether normal or anomaly, in real-time. Figure 3 summarizes the whole process.

$$f^{in}_{s,sp,i,ip}(t) = \frac{|F^{t,\delta t}_{in}(s,sp,i,ip)|}{\delta t} \qquad f^{out}_{i,ip,d,dp}(t) = \frac{|F^{t,\delta t}_{out}(i,ip,d,dp)|}{\delta t},$$

$$f^{in}_{s,i,ip}(t) = \frac{|\bigcup_{\forall sp} F^{t,\delta t}_{in}(s,sp,i,ip)|}{\delta t} \qquad f^{out}_{i,ip,d}(t) = \frac{|\bigcup_{\forall dp} F^{t,\delta t}_{out}(i,ip,d,dp)|}{\delta t},$$

$$f^{in}_{s,sp,i}(t) = \frac{|\bigcup_{\forall ip} F^{t,\delta t}_{in}(s,sp,i,ip)|}{\delta t} \qquad f^{out}_{i,d,dp}(t) = \frac{|\bigcup_{\forall ip} F^{t,\delta t}_{out}(i,ip,d,dp)|}{\delta t},$$

$$f^{in}_{s,i}(t) = \frac{|\bigcup_{\forall sp,ip} F^{t,\delta t}_{in}(s,sp,i,ip)|}{\delta t} \qquad f^{out}_{i,d}(t) = \frac{|\bigcup_{\forall ip,dp} F^{t,\delta t}_{out}(i,ip,d,dp)|}{\delta t},$$

$$f^{in}_{sp,i}(t) = \frac{|\bigcup_{\forall s,ip} F^{t,\delta t}_{in}(s,sp,i,ip)|}{\delta t} \qquad f^{out}_{i,dp}(t) = \frac{|\bigcup_{\forall ip,d} F^{t,\delta t}_{out}(i,ip,d,dp)|}{\delta t},$$

$$f^{in}_{i,ip}(t) = \frac{|\bigcup_{\forall s,sp} F^{t,\delta t}_{in}(s,sp,i,ip)|}{\delta t} \qquad f^{out}_{i,ip}(t) = \frac{|\bigcup_{\forall d,dp} F^{t,\delta t}_{out}(i,ip,d,dp)|}{\delta t},$$

$$f^{in}_{i}(t) = \frac{|F^{t,\delta t}_{in}(i)|}{\delta t} \qquad f^{out}_{i}(t) = \frac{|F^{t,\delta t}_{out}(i)|}{\delta t},$$

$$maxf^{in}_{i}(t) = \max_{ip}\{f^{in}_{i,ip}(t)\}. \qquad max_{ip}\{f^{out}_{i,ip}(t)\}.$$

Fig. 2 In Frequency number & out frequency number [2]

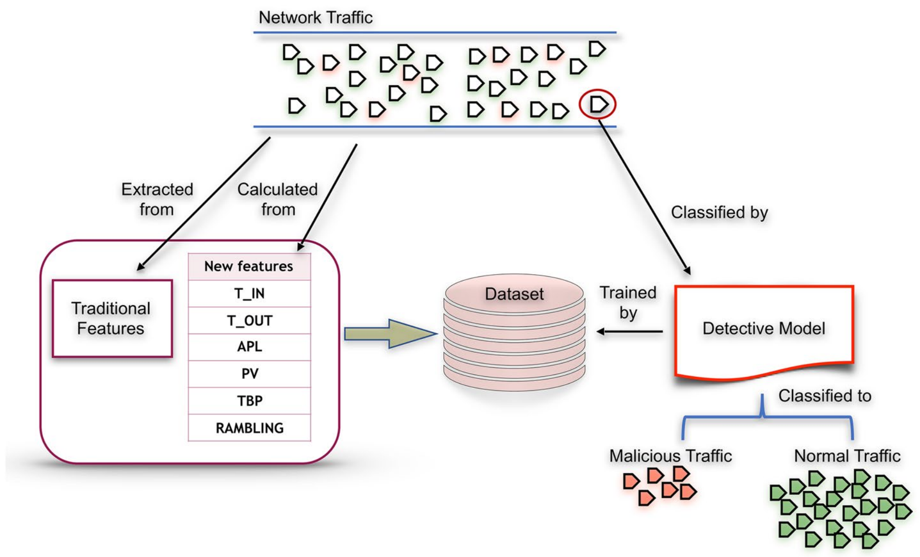

Fig. 3 Detection process

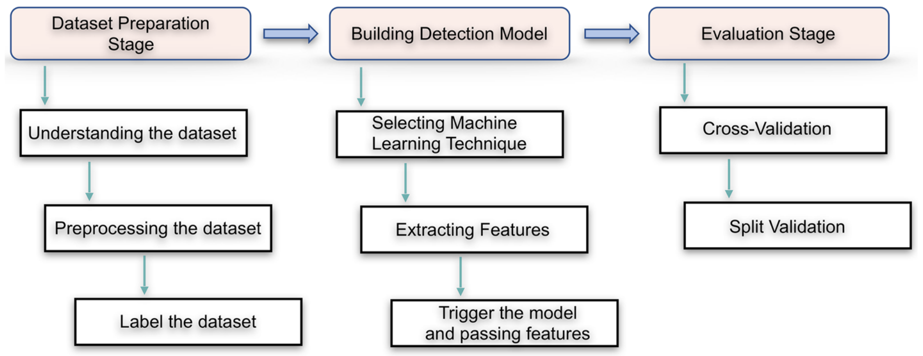
Dataset Preparation Stage
Building Detection Model
Evaluation Stage
Understanding the dataset
Preprocessing the dataset
Label the dataset
Selecting Machine Learning Technique
Extracting Features
Trigger the model and passing features
Cross-Validation
Split Validation

Fig. 4 Flowchart showing method in the research

Methods

The methodology of our work illustrated in Fig. 4. It consists of three stages. Stage 1 concerns the dataset preparation, and stage 2 builds the detection model. The last stage will consist of the evaluation stage, which ensures our approach accuracy for anomaly detection.

Dataset preparation stage

Understanding dataset

Cloud computing networks facing security threats, same as the traditional computing networks with some other differences [1]. According to several protocols, services, and technologies such as virtual structures, these additional security threats related to the cloud infrastructure have data formatting levels. With such an environment providing protection should consider all data traffic in both insider and outsider. The remaining challenge of completing this job is building an ML model that trains IDS to capture these various data abstraction anomalies. Furthermore, the extracting features from these several data places need related tools to pass the gathered row data to the trained ML model. The extracting tools should be gathering recent instances of data from several resources in real-time.

ISOT-CID [1] dataset was presented as an exciting job contains several data collections about data transmission behavior and buffer data format. The presented dataset has enough properties and data attributes to train IDS for robust and comprehensive protection. The data collections of ISOT-CID consist of system call properties, network traffic memory dump, events log, and resource utilization. The ISOT-CID cloud intrusion detection dataset contains terabytes of data, including regular traffic, activities, and multiple attack scenarios. The data gathered in several periods in the cloud in a natural environment. This dataset's content is considered essential for the business industry for developing a realistic intrusion detection model for cloud computing. The ISOT dataset collects various data goatherd from cloud environment and collected from different cloud layers, involved guest hosts, networks, and hypervisors,

Table 1 The feature extracted from network traffic

Features	Description
S_MAC	Source MAC Address in the data link frame
D_MAC	Destination MAC address in the data line frame
S_IP	Source IP of the data packet
S_PT	Source Port of the data packet
D_IP	Destination IP of the data packet
D_PT	Destination Port of the data packet
IP_LEN	The length of the IP packet
IP_HLEN	The IP header length of the IP packet
TCP_HLEN	The TCP header length
FR_LEN	The packet's length includes the header and data identification for the current data in a packet
IP_OFFS	The offset of the IP packet
TCP_SEQ	Data location of the TCP segment
TCP_ACK	Number of data received

Table 2 Extracted features (calculated) of interval time

Features	Description
T_IN	A frequency number. The count of the appearing packets incoming to the destination IP address throughout the observation time window
T_OUT	A frequency number. The count of the appearing packets is outgoing from the destination IP address throughout the observation time window
APL	Average payload packet length for the time interval
PV	A variance of payload packet length for the time interval
TBP	The average time between packets in a time interval
RAMBLING	The rambling amount of payload packet length means in a time interval

and encompasses data with various data formats and several data resources such as memory, CPU, system, and network traffic. It includes various attack scenarios such as a denial-of-service masquerade attack, stealth attacks, attacks data from inside and outside the cloud, and anomalous user behavior. ISOT-CID aims to represent a real dataset in the cloud for scientist's researchers so that they can develop, evaluate and make a comparison of their works. It intends to help various and comprehensive IDS systems development and evaluation.

Furthermore, ISOT-CID is fundamentally raw data and has not been converted, altered, or manipulated. It is prepared and structured for securing the cloud community. In this research, we consider only the network traffic part, as described in the Ph.D. thesis of Aldribi et al. [1].

In this research, we are working on only the network traffic part. The dataset attributes describe in Tables 1 and 2.

Preprocessing the dataset

The preprocessing dataset means looking for data instances to deduct redundancy, handle missing values and outlier values. Most ML algorithms need data organized in a way that is suitable to their procedure. So, datasets demand preparation and preprocessing before they can produce valuable patterns. Usually, datasets have missing and invalid data or otherwise difficult for an algorithm to process.

If data is missing, the algorithm cannot deal with it. If data is invalid, the algorithm produces less accurate outcomes. As preprocessing, we convert the columns protocol, MAC source, and MAC destination from categorical data to be numeric to fed into the machine learning algorithm. The conversion process is done by Python code and related libraries.

The dataset we arranged to consist of 416 dump files contains network traffic flow extracted from networks in several periods. The dataset contains only no calculated attributes described in Table 1. We use WIRE- SHARK Version 1.10.2 to extract these features from dump files and save them in corresponding CSV files.

The calculated attributes illustrated in Table 2 in the previous section. These attributes compute by the Java program designated for this purpose. This Java program uses 0,003 as interval time to compute most of the attributes according to their formula. Our contributed feature, which is called RAMBLING, is computed with the same interval time. The last attribute is label class, as described in the next section. The total size of the dataset, which contains all attributes consist of 89,364 instances.

Label the dataset

Labeling dataset is a significant process for training the ML Algorithm to classify the new traffic as malicious or normal. After computing the attributes in Table 2 in the previous section using the Java program, we extend the program for labeling the instance class by Normal if it has a source or destination IP address. The list of Normal IP addresses shown in Table 3 otherwise Malicious. The java program produces only 1612 instances as malicious and 87,752 instances as normal. The anomaly numbers of instance founded in the dataset is good, but this gives the dataset are imbalanced. To preserve the Normal instance number is large enough and increase the number of malicious, we use over-sampling and under-sampling methods to make a balanced dataset containing 44,569 instances as Malicious and 44,795 Normal instances. The total number of instances in the dataset used in the training ML model is 89364 instances.

We believe that the over-sampling and under-sampling change in misclassification costs and class distribution. Also, over-sampling is unexpectedly effective and producing a change in performance. However, it is noteworthy that representing these changes internally by down-weighting gives the best performance overall [7]. In our dataset, we experimented before under-sampling and over-sampling. The result found in cross-validation some folds give low accuracy while the average accuracy is still high. The two columns feature used for labeling removed from the dataset, so the ML models are trained by others, which are (Time -Protocol–Length -Source Port-Destination Port-IPHdrLength-SOURCE MAC-DMAC-TCPHdr-Length-FramLength-IPOfsetNo-TCPSEQ-TCP_ACK-F_IN-F_out-Rambling-APL-PV-TBP-class).

Table 3 Source IP address

Source ip address
142.104.64.202
172.16.1.21
172.16.1.255
172.16.1.24
172.16.1.20
172.16.1.23
172.16.1.10
172.16.1.28
172.16.1.26
172.16.1.19
172.16.1.27
192.168.0.10
192.168.0.8
192.168.0.9
255.255.255.255
142.104.64.196
142.104.80.2
142.104.6.1
134.87.154.134
134.87.157.129
142.104.191.194
206.12.96.150
206.12.96.149
206.12.96.143
206.12.96.142
206.12.96.141
206.12.96.239
206.12.96.240
206.12.59.162
206.12.48.43
172.16.1.254
172.16.1.17
172.16.1.16
206.12.48.115
2.89.202.164
206.87.164.188
51.218.248.166
188.49.184.140
220.54.25.114
37.105.194.65
5.156.104.155
51.36.227.160
151.255.23.160
94.48.53.15

Building detection model stage

Selecting ML technique

In this task, we construct a model with several well-known ML models for selecting the accurate classifier. These well-known models are the Decision Tree (DT), Neural Networks (NNs), K-nearest neighbor (KNN), Naïve Bayes (NB), Support Vector Machine (SVM), and Finally, Random Forest (RF).

Extracting features

After building, testing, and evaluating the detection model, this task can use when the system is deployed and fitted into the memory. In real-time, one by one feature can extract from network flow traffic. For those interested in our result, can for dataset feature and ML experiments use our approach to create full software to be a feeder of the IDS system on the computer network.

Trigger the model and passing features

After the feature extracted in Real-Time should pass into the detection model for classification, IDS can alert for another device for decision-making once the packet is classified.

Evaluating stage

Cross-validation

In this task, we conduct several experiments to evaluate the ML algorithm for accuracy. The confusion matrix uses to calculate the percentage of accuracy of each algorithm.

Split -validation

The alternative technique also uses for judgment of the accuracy of the ML model. It split the data into training and testing parts 90%, 80%, or 70%; the testing part uses to calculate the presence of each algorithm's accuracy by using the confusion matrix.

Results and analysis

The result of training ML models by provided dataset which described in the previous section is consists of two sections as the following:

Cross-validation evaluation

Cross-validation is a technique used to validate the ML algorithm according to divide the dataset into folds to ensure all kinds of dataset instances hold in training and testing. This division is called K-folds, where K represents the number of division parts. For example, K-folds = 5 means the dataset split into five parts, where part-1 uses for training and part-2 for testing as fold-1. In fold-2, part-2 takes as training and part-3 for testing. Part-3 uses in training and part-4 for testing in fold-3. Fold-4 gives part-4 for training and part-5 testing. In fold-5, part-5 uses for training, and part-1 for testing at the end. The model accuracy is the average accuracy of all five folds. This

Table 4 ANN model accuracy result

Value of K fold	Accuracy (/1)
5	0.92
10	0.94
15	0.92

Table 5 DTREE model accuracy result

Value of K fold	Accuracy (/1)
5	1.00
10	1.00
15	1.00

Table 6 KNN model accuracy result

Value of K fold	Accuracy (/1)
5	1.00
10	1.00
15	1.00

technique will ensure if there is overfitting in training or not. The meaning of overfitting in machine learning is there is no clear separation in data instances by other meaning the value of the attributes are closer so; ML could take the same instance in the classes.

Evaluating ANN

Table 4 illustrates three experiments result for the ANN model. The first one by using K = 5, K = 10 in the second, and K = 15 in the third experiment. This result shows that most folds accuracy is closer, which ensures no overfitting in the ML model, and the accuracy result is 94% which is acceptable.

Evaluating DTREE

Table 5 shows three experimental results for the DTREE model with K = (5, 10 and 15). The accuracy result given is unexpected, but it comes as 100%.

Evaluating K-nearest Neighbor (KNN classifier)

Table 6 illustrates that the KNN model is also applicable to be reliable for detecting anomalies by the presented dataset.

Evaluating support vector machine

Table 7 presents that the SVM model is not appropriate for detecting anomalies by a presented dataset.

Table 7 SVM model accuracy result

Value of K fold	Accuracy (/1)
5	0.68
10	0.81
15	0.84

Table 8 Random Forest model accuracy result

Value of K fold	Accuracy (/1)
5	1.00
10	1.00
15	1.00

Evaluating Random Forest

Table 8 shows that the Random Forest model and Decision Tree give the same result, which is 100%.

Evaluating Naive Bayes

Table 9 shows that the Naïve Bayes model is not applicable to be reliable for detection anomaly by a presented dataset. The model gives a pure result with cross-validation among three experiments in different kinds of folds.

Split-validation evaluation

This evaluation method breaks apart from dataset instances for testing after fitting the ML model running in the memory and trained by another. That is means dividing the dataset into two parts, one for testing and the other for training. The accuracy of the model is given by computing the confusion matrix that consists of four values:

- True Positive (TP): This is the number of observations positive and predicted to be positive.
- True Negative (TN): This is the number of observations positive and predicted to be negative
- False Positive (FP): This is the number of observations negative but predicted to be positive.
- False Negative (FN): This is the number of observations positive but predicted to be negative.

Table 9 Naïve Baye model accuracy result

Value of K fold	Accuracy (/1)
5	0.60
10	0.60
15	0.60

Table 10 Confusion matrix of ANN

	Predicted normal	Predicted malicious
Actual normal	TP (39,130)	FP (931)
Actual malicious	FN (2533)	TN (37,834)

Evaluating ANN

Tables 10 and 11 show the ANN model's accuracy result, which is 0.96, according to the split-validation evaluation technique. In this experiment, we use 90% of data instances for training and 10% for testing. The Confusion Matrix presented in Table 10 clarifies that 39,130 instances classify as normal from the testing data part, where they label as expected in the dataset. The classifier ANN failed with 931 instances where these instances were labeled normal in the dataset and classified as Malicious is the wrong classification. On the other hand, 37,834 instances classify by ANN as accurate as malicious, where 2533 instances are classified as usual as wrong, while the ANN classifier should classify them as malicious. The total accuracy result is 0.96% is acceptable and can be reliable to feed the IDS for anomaly detection.

Evaluating DTREE

Tables 12 and 13 present the DTREE model result, which is 100% according to training by 90% of the dataset tested by 10% of dataset instances. The confusion matrix illustrated in Table 12 clarifies that no wrong instance was found in the testing part after classification Table 13.

Table 11 Classification report of ANN

Class	Precision	Recall	F1-score
Normal	0.94	0.98	0.96
Malicious	0.98	0.94	0.96

Table 12 Confusion matrix of DTREE

	Predicted normal	Predicted malicious
Actual normal	TP (40,061)	FP (0)
Actual malicious	FN (0)	TN (40,367)

Table 13 Classification report of DTREE

Class	Precision	Recall	F1-score
Normal	1.00	1.00	1.00
Malicious	1.00	1.00	1.00

Evaluating KNN

As in Table 14 for K-nearest Neighbor (KNN) model, the confusion matrix clarifies that 489 instances failed in a classification where 39,572 instances have a correct classification in the Normal class. On the other side,362 have classification errors as Normal where these instances should be malicious, and 40,005 instances classified true as malicious. Also, the classification report presents in Table15.

Evaluating SVM

Tables 16 and 17 SVM give an 81% accuracy result by splitting the dataset into 90% for training and 10% for testing.

Evaluating Random Forest

Tables 18 and 19 presented that Random Forest is the most accurate model, same as DTREE for anomaly detection in network traffic flow.

Table 14 Confusion Matrix of KNN

	Predicted normal	Predicted malicious
Actual normal	TP (39,572)	FP (489)
Actual malicious	FN (362)	TN (40,005)

Table 15 Classification report of KNN

Class	Precision	Recall	F1-score
Normal	0.99	0.99	0.99
Malicious	0.99	0.99	0.99

Table 16 Confusion matrix of SVM

	Predicted normal	Predicted malicious
Actual normal	TP (40,061)	FP (0)
Actual malicious	FN (15,554)	TN (24,813)

Table 17 Classification report of SVM

Class	Precision	Recall	F1-score
Normal	0.72	1.00	0.84
Malicious	1.00	0.61	0.76

Table 18 Confusion matrix of Random Forest

	Predicted normal	Predicted malicious
Actual normal	TP (40,061)	FP (0)
Actual malicious	FN (0)	TN (40,367)

Table 19 Classification report of Random Forest

Class	Precision	Recall	F1-score
Normal	1.00	1.00	1.00
Malicious	1.00	1.00	1.00

Table 20 Confusion matrix of Naïve Bayes

	Predicted normal	Predicted malicious
Actual normal	TP (7786)	FP (32,275)
Actual malicious	FN (0)	TN (40,367)

Table 21 Classification report of Naïve Bayes

Class	Precision	Recall	F1-score
Normal	1.00	0.19	0.33
Malicious	0.56	1.00	0.71

Evaluating Naive Bayes

Tables 20 and 21 show that the Naïve Bayes model is not applicable for prediction anomaly, where it has low accuracy of 60%.

Discussion

We get good results by conducting several experiments by python programming language on the ISOT-CID dataset, collected from network traffic extracted in different periods. Six ML models are trained by this dataset and evaluated by two evaluation methods cross-validation and split-validation. Four of them give significant accurate result while the other two give none accepted result as the following:

Cross-validation result

The evaluation method cross-validation is conducted several times with different values of K-Fold on the dataset. Table 22 shows the result of each experiment for a specific ML model. Also, Fig. 5 visualizes the result of each experiment on the dataset. Cross-validation gave the same result by split-validation, DTREE and Random Forest produces an optimal result with no error or mistake found in the testing fold allocated from the dataset. That is means DTREE, and Random Forest models are most accurate and applicable to be a feeder for IDS to anomaly detection on network traffic flow.

Split-validation result

By conducting an ML experiment on the IOST-CID dataset. Table 23 and Fig. 6 show six ML model results. DTREE and Random Forest gave optimal results 100%, which means

Table 22 Cross-validation result for ML models

ML model	K-folds	Accuracy (%)
ANN	5	92
ANN	10	94
ANN	15	92
KNN	5	100
KNN	10	100
KNN	15	100
DTREE	5	100
DTREE	10	100
DTREE	15	100
SVM	5	68
SVM	10	81
SVM	15	84
Naïve Bayes	5	60
Naïve Bayes	10	60
Naïve Bayes	15	60
Random Forest	5	100
Random Forest	10	100
Random Forest	15	100

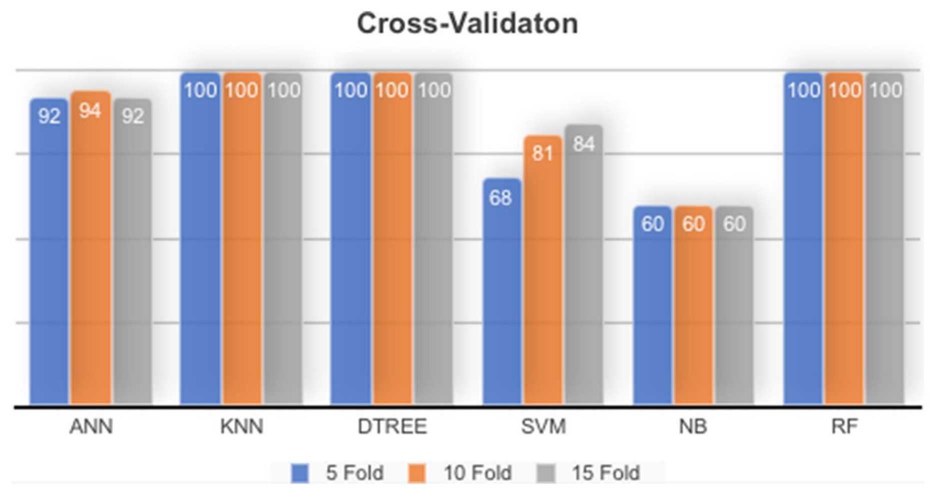

Fig. 5 Model accuracy comparison with cross-validation

no error or mistake was found in the classification process on the testing part allocated from the dataset.

While all the results are excellent, random forest and DTREE show that they give the best results based on split validation or cross-validation. We think that this due to the characteristics of random forest and DTREE. Random forest characteristics are:

Table 23 Split-Validation result for ML models

ML model	Splitting (%)	Accuracy (%)
ANN	90	96
ANN	80	93
ANN	70	97
KNN	90	100
KNN	80	99
KNN	70	100
DTREE	90	100
DTREE	80	100
DTREE	70	100
SVM	90	68
SVM	80	81
SVM	70	84
Naïve Bayes	90	60
Naïve Bayes	80	60
Naïve Bayes	70	60
Random Forest	90	100
Random Forest	80	100
Random Forest	70	100

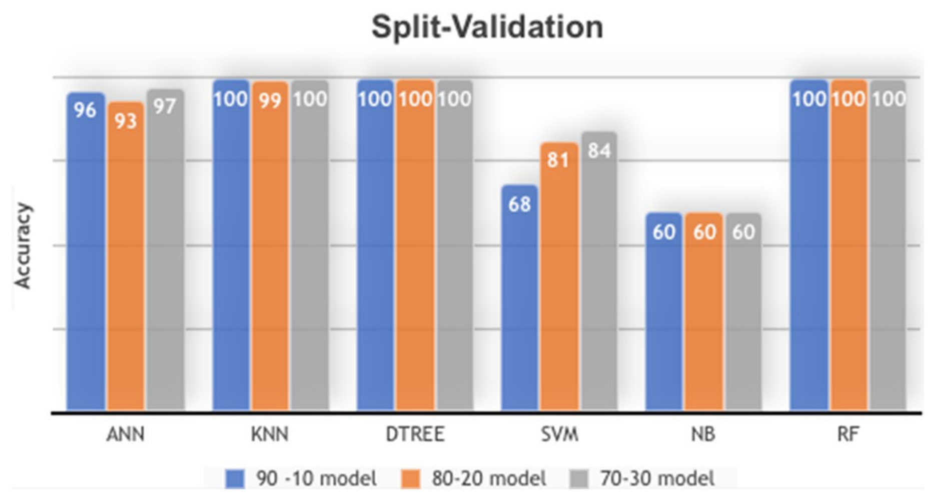

Fig. 6 Model accuracy comparison with split-validation

- It needs to be some actual signal in the dataset features, which helps it do better. It is available in our dataset.
- The predictions (and therefore the errors) made by the individual trees need to have low correlations.
- While the DTREE explores all possible outcomes of a decision, this methodology helps create an analysis that includes all the outcomes. It is suitable for our comprehensive dataset.

Conclusions and future work

As one of the extensive uses of computer networks and telecommunication devices, network security has become significant for all these networks' users. Consequently, this issue of intrusion detection has identified the helpfulness of both research and corporate associations intending to develop and deploy effective IDSs that are proficient in protecting severe system components against intruders.

We present a reliable model running in Real-time to detect malicious data flow traffic depending on the ML supervised techniques based on the ISOT-CID dataset that contains network traffic data features. Our challenge in this research is to capture the deviations between the data instances so; malicious and normal properties categorize the data. Six column features are computed and added to the network traffic properties to support the ML model for diagnoses the malicious traffic.

We present one novel feature called rambling that compute related to interval time of traffic data connection. The packet payload length can be extracted at this period and compute the diversion of length about the mean of all packet's length. We approved that the six features added to the dataset are vital to producing a qualitative dataset applicable to the train machine learning model for anomaly detection. DTREE and Random Forest are both gave optimal accuracy results when evaluated by cross-validation and split-validation. These two models did not fail in any instance on the classification process applied to testing parts or folds from the dataset.

Despite the encouraging results of the machine learning models are used and the six vital features that have been able to raise the efficiency of machine learning models, there are some limitations in the model presented. IDS security systems for computer networks must be very fast where it is deployed in real-time to extract the communication traffic characteristics and give its response in real-time. The presented model relies on a vast dataset that is considered a type of big data, where it influenced the performance of fitting the system and its evaluation. Simultaneously, the deployment of this model in real networks will harm the speed required. Therefore, we will apply deep learning techniques using cloud computing to exploit the dataset, integrating with the six calculated features as future work.

Acknowledgements
I would like to state my special thanks and appreciation to Jouf university that gave me the best chance to study for my master's degree.

Authors' contributions
AAls and AAld have participated in the design of the proposed method. AAls has implemented and coded the method and go testing and obtain the results. As a supervisor, AAld support and guide AAls during her MSc degree with some ideas and knowledge. Both authors read and approved the manuscript.

Funding
The corresponding author has a master's degree funding from Jouf University.

Availability of data and materials
The data that used in this research are available upon request from the author Dr. Abdulaziz Aldribi. Data are publicly available and available under this website https://www.uvic.ca/engineering/ece/isot/datasets/cloud-security/index.php.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests

I declare that "on behalf of all authors" I have no significant competing financial, professional, or personal interests that might have influenced the performance or presentation of the work described in this manuscript.

Author details

[1]Department of Computer Science, College of Computer, Jouf University, Al Jouf, Saudi Arabia. [2]Department of Computer Science, College of Computer, Qassim University, Buraydah, Saudi Arabia.

Received: 4 September 2020 Accepted: 24 May 2021
Published online: 14 June 2021

References

1. Aldribi A, Traore I, Moa B. Data sources and datasets for cloud intrusion detection modeling and evaluation. In: Cloud computing for optimization: foundations, applications, and challenges. Cham: Springer; 2018. p. 333–66.
2. Aldribi A, Traoré I, Moa B, Nwamuo O. Hypervisor-based cloud intrusion detection through online multivariate statistical change tracking. Comput Secur. 2020;88:101646–101646. https://doi.org/10.1016/j.cose.2019.101646.
3. Anton SD, Kanoor S, Fraunholz D, Schotten HD. Evaluation of machine learning-based anomaly detection algorithms on an industrial modbus/tcp data set. In:Proceedings of the 13th international conference on availability, reliability and security. 2018. p. 1–9.
4. Barbhuiya S, Papazachos Z, Kilpatrick P, Nikolopoulos DS. RADS: Real-time anomaly detection system for cloud data centres. 2018. arXiv preprint arXiv:1811.04481.
5. Bellaïche M, Grégoire JC. SYN flooding attack detection by TCP handshake anomalies. Secur Commun Netw. 2012;5(7):709–24. https://doi.org/10.1002/sec.365.
6. Desai MM. Hacking for beginners: a beginners guide to learn ethical hacking. 2010.
7. Drummond C, Holte RC. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In: Workshop on learning from imbalanced datasets II, vol. 11. Washington DC: Citeseer; 2003. p. 1–8.
8. Fernández GC, Xu S. A case study on using deep learning for network intrusion detection. In: MILCOM 2019–2019 IEEE Military Communications Conference (MILCOM). IEEE; 2019. p. 1–6.
9. Garg S, Kaur K, Kumar N, Kaddoum G, Zomaya AY, Ranjan R. A hybrid deep learning-based model for anomaly detection in cloud datacenter networks. IEEE Trans Netw Serv Manage. 2019;16(3):924–35. https://doi.org/10.1109/tnsm.2019.2927886.
10. Gelman D, Shvartsev B, Ein-Eli Y. Aluminum–air battery based on an ionic liquid electrolyte. J Mater Chem A. 2014;2(47):20237–42. https://doi.org/10.1039/c4ta04721d.
11. Jianliang M, Haikun S, Ling B. The application on intrusion detection based on k-means cluster algorithm. In: 2009 International Forum on Information Technology and Applications, vol. 1. IEEE; 2009. p. 150–2.
12. Kumari R, Singh MK, Jha R, Singh NK. Anomaly detection in network traffic using K-mean clustering. In: 2016 3rd International Conference on Recent Advances in Information Technology (RAIT). IEEE; 2016. p. 387–93.
13. Kwon D, Kim H, Kim J, Suh SC, Kim I, Kim KJ. A survey of deep learning-based network anomaly detection. Cluster Comput. 2019;22(1):949–61.
14. Lemay A, Fernandez JM). Providing {SCADA} network data sets for intrusion detection research. In: 9th Workshop on Cyber Security Experimentation and Test ({CSET} 16). 2016.
15. Manna A, Alkasassbeh M. Detecting network anomalies using machine learning and SNMP-MIB dataset with IP group. In: 2019 2nd International Conference on new Trends in Computing Sciences (ICTCS). IEEE; 2019. p. 1–5.
16. Peel D, McLachlan GJ. Robust mixture modelling using the t distribution. Stat Comput. 2000;10(4):339–48.
17. Mobilio M, Orrù M, Riganelli O, Tundo A, Mariani L. Anomaly detection as-a-service. In: 2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). IEEE; 2019. p. 193–9.
18. Moustafa N, Creech G, Sitnikova E, Keshk M. Collaborative anomaly detection framework for handling big data of cloud computing. In: 2017 military communications and information systems conference (MilCIS). IEEE; 2017. p. 1–6.
19. Münz G, Li S, Carle G. Traffic anomaly detection using k-means clustering. In: GI/ITG Workshop MMBnet. 2007. p. 13–4.
20. Nikiforov R. Clustering-based anomaly detection for microservices. 2018. arXiv preprint arXiv:1810.02762.
21. Nisioti A, Mylonas A, Yoo PD, Katos V. From intrusion detection to attacker attribution: a comprehensive survey of unsupervised methods. IEEE Commun Surv Tutor. 2018;20(4):3369–88. https://doi.org/10.1109/comst.2018.2854724.
22. Osanaiye O, Cai H, Choo KKR, Dehghantanha A, Xu Z. Dlodlo M (2016) Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing. EURASIP J Wirel Commun Netw. 2016;1:130–130. https://doi.org/10.1186/s13638-016-0623-3.
23. Parul B, Kumar Gurjwar R. A review on attacks classification using decision tree algorithm. Int J. 2014;2(2).
24. Peng K, Leung VCM, Zheng L, Wang S, Huang C, Lin T. Intrusion detection system based on decision tree over big data in fog environment. Wirel Commun Mob Comput. 2018;2018:1–10. https://doi.org/10.1155/2018/4680867.
25. Pilli ES, Joshi RC, Niyogi R. Data reduction by identification and correlation of TCP/IP attack attributes for network forensics. In: Proceedings of the International Conference & Workshop on Emerging Trends in Technology. 2011. p. 276–83.
26. Qiu Z, Miller DJ, Kesidis G. Detecting clusters of anomalies on low-dimensional feature subsets with application to network traffic flow data. In: 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP). IEEE; 2015. p. 1–6.
27. Umer MF, Sher M, Bi Y. Flow-based intrusion detection: Techniques and challenges. Comput Secur. 2017;70:238–54. https://doi.org/10.1016/j.cose.2017.05.009.

28. Zhang J. Anomaly detecting and ranking of the cloud computing platform by multi-view learning. Multimedia Tools Appl. 2019;78:30923–42.
29. Zhao D, Traore I, Sayed B, Lu W, Saad S, Ghorbani A, Garant D. Botnet detection based on traffic behavior analysis and flow intervals. Comput Secur. 2013;39:2–16. https://doi.org/10.1016/j.cose.2013.04.007.

Publisher's Note