Array 22 (2024) 100349

Contents lists available at ScienceDirect

Array

journal homepage: www.elsevier.com/locate/array

Network intrusion detection leveraging multimodal features

Aklil Kiflay *, Athanasios Tsokanos, Mahmood Fazlali, Raimund Kirner

University of Hertfordshire, Hatfield, AL10 9AB, United Kingdom

ARTICLE INFO

Keywords:
Intrusion detection
Network flow
Packet payload
Random forest
Machine learning

ABSTRACT

Network Intrusion Detection Systems (NIDSes) are essential for safeguarding critical information systems. However, the lack of adaptability of Machine Learning (ML) based NIDSes to different environments could cause slow adoption. In this paper, we propose a multimodal NIDS that combines flow and payload features to detect cyber-attacks. The focus of the paper is to evaluate the use of multimodal traffic features in detecting attacks, but not on a practical online implementation. In the multimodal NIDS, two random forest models are trained to classify network traffic using selected flow-based features and the first few bytes of protocol payload, respectively. Predictions from the two models are combined using a soft voting approach to get the final traffic classification results. We evaluate the multimodal NIDS using flow-based features and the corresponding payloads extracted from Packet Capture (PCAP) files of a publicly available UNSW-NB15 dataset. The experimental results show that the proposed multimodal NIDS can detect most attacks with average Accuracy, Recall, Precision and F_1 scores ranging from 98% to 99% using only six flow-based traffic features, and the first 32 bytes of protocol payload. The proposed multimodal NIDS provides a reliable approach to detecting cyber-attacks in different environments.

1. Introduction

Network Intrusion Detection Systems (NIDSes) are used to detect security threats to information systems [1,2]. To detect attacks effectively, state-of-the-art NIDSes have used Machine Learning (ML) [3–9]. Broadly, NIDSes are either *signature-based* or *anomaly-based* systems [10]. Signature-based NIDSes detect intrusions by comparing network traffic with known attack patterns using a set of rules predetermined by security experts. In contrast, anomaly-based NIDSes use ML algorithms to model normal traffic behavior and deviations thereof are considered as attacks. While ML has been applied in both signature-based and anomaly-based intrusion detection, most ML-based NIDSes nowadays use anomaly-based approach. With respect to what aspect of a network packet is analyzed for intrusion detection, ML-based NIDSes can be further classified into *flow-based* and *payload-based*.

Flow-based NIDSes use statistical features and packet metadata, collectively referred to as flow-based features, to identify unexpected changes in network traffic and detect attacks. As they use summarized information, flow-based NIDSes are more scalable and require less computing resources than payload-based methods. However, flow-based NIDSes have limitations that could hamper their adoption in practice. Firstly, since they do not analyze user data in network packets, flow-based NIDSes lack capability to detect threats concealed within packet payloads. Secondly, flow-based NIDSes are not usually domain-adaptive because the features based on which they detect attacks can change across networks. Thirdly, there is no consensus among researchers on which flow-based features to use in ML-based NIDSes, resulting in lack of standardization, compatibility, repeatability, and adaptability to different domains and environments [11,12]. These problems have led to flow-based NIDSes that can detect attacks accurately in one network environment, while giving unacceptably low levels of detection in others [13,14]. Furthermore, extracting and pre-processing flow-based features require human expertise and tend to be costly and time-consuming.

Unlike their flow-based counterparts, payload-based NIDSes detect attacks using the actual user data exchanged between hosts. While payload-based NIDSes inspect payloads of network packets, they do not use flow-based features for intrusion detection. Instead, packet contents are analyzed including any application data. Payload-based methods are particularly useful for detecting attacks that are embedded as user contents. Therefore, payload-based NIDSes are effective in detecting various threats such as application-layer attacks. However, payload-based NIDSes require large computational resources and have limited scalability.

In this paper we focus on an NIDS for offline attack detection, i.e., we focus on after-the-fact detection of attacks. This decision is noteworthy, as it allows us to access information that would not be available to an online NIDS for intrusion prevention. For example, this

* Corresponding author.
E-mail addresses: a.z.kiflay@herts.ac.uk (A. Kiflay), a.tsokanos@herts.ac.uk (A. Tsokanos).

https://doi.org/10.1016/j.array.2024.100349
Received 7 November 2023; Received in revised form 5 February 2024; Accepted 14 May 2024
Available online 16 May 2024

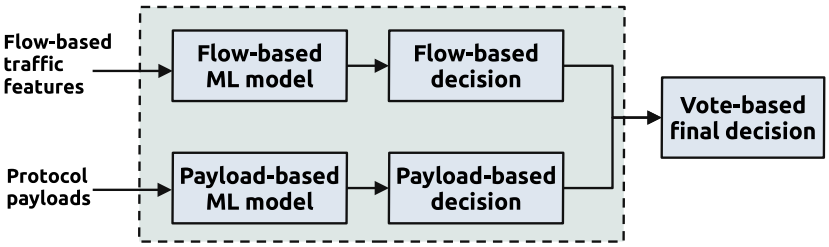

Fig. 1. Multimodal ML-based NIDS.

allows us to access statistical features that are computed after the end of the traffic flow.

Our NIDS approach is based on machine learning and uses a multimodal detection using flow-based and payload-based information, which is shown in Fig. 1. We use a soft-voting approach to merge the predictions of the two detection models.

This paper provides the following scientific contributions:

1. Proposing a multimodal NIDS using flow-based and payload-based information (see Fig. 1). To the best of our knowledge, this is the first *ML-based NIDS* approach that combines flow-based and payload-based information as a *multimodal* approach. To merge the results of the two models, we have used a soft-voting method.
2. Focus on first 32 bytes of protocol payload to minimize payload processing overhead of the NIDS. Previous works focused on larger packet payload sizes.
3. An experimental evaluation of the multimodal NIDS.
4. An algorithm to extract and label the protocol payload from the traffic data. This algorithm is needed, as publicly available open source datasets provide labeled flow-based traffic features, but not the extracted protocol payload information.

The paper is structured as follows: Section 2 reviews related work on flow-based and payload-based intrusion detection. The proposed multimodal NIDS is presented in Section 3. Section 4 presents evaluation of the proposed method. In Section 5, the obtained results are presented and comparisons with alternative approaches are made. Section 6 concludes the paper.

2. Related work

In this section, previous studies are classified and discussed. With respect to data used to detect attacks, the broad categories of approaches are flow-based and payload-based methods. While packet metadata from protocol headers and traffic statistics are used in flow-based techniques, payload-based methods use either the actual user data of application-layer protocols or features extracted thereof as a foundation for detecting intrusions [15,16].

2.1. Flow-based intrusion detection

In flow-based methods, cyber-attacks are identified by tracking changes in the statistical properties of network traffic and packet metadata, collectively referred to as flow-based features. Several ML-based NIDSes monitor flow-based features to detect cyber-attacks [17–22]. Therefore, a significant body of previous work in ML-based NIDSes focused on determining a subset of flow-based features that result in high detection rates [2,23–26]. However, NIDSes that are based on flow features usually consume high dimensional data as input. As a result, they tend to require high computational resources due to the curse of dimensionality [27,28].

Consequently, ML-based NIDSes that depend on flow-based features have limitations that hamper their adoption in practice. Firstly, many of the statistical traffic attributes are not domain-adaptive because features using which an ML classifier can detect attacks in one network may not enable the detection of the same attacks in another network due to inherent unpredictability of network traffic [29,30]. Secondly, there is no consensus among researchers on which set of traffic features to use in ML-based NIDSes [31]. As a result, current ML-based NIDSes use non-standard set of traffic features, creating compatibility and adaptability problems across networks [32,33].

Overall, most ML-based NIDSes tend to use flow-based traffic features for attack detection, thereby focusing more on statistical attributes of network traffic rather than packet payloads. Flow-based NIDSes have minimum data processing costs because they use summarized traffic information. However, they can detect only a limited range of cyber-attacks since they rely solely on information extracted from packet headers. Flow-based methods do not scan packet payloads, and therefore, their capacity to detect network attacks embedded within application-layer data is relatively lower than payload-based methods [34].

2.2. Payload-based intrusion detection

Some ML-based NIDSes detect cyber-attacks based on the actual user data exchanged during network traffic transactions [35]. In payload-based intrusion detection, data from a network application is analyzed for signs of maliciousness. Previously, some payload-based NIDSes have been proposed. In general, most of these NIDSes are designed to classify traffic using either *N-gram* analysis of the payload or *deep learning* methods.

2.2.1. N-gram analysis

N-grams are used to model the payload distributions of normal and attack traffic. Although N-gram methods were originally developed for text categorization [36], they have been applied in network traffic classification and intrusion detection [37–39]. To analyze traffic using N-grams, packet payloads are treated as strings of bytes. From each packet, sets of payload sub-strings of a fixed length are extracted and analyzed to identify attacks. Accordingly, by taking the user content as the basis of attack detection, a number of payload-based NIDSes extracted features using N-grams [40–42].

2.2.2. Deep learning approaches

Deep neural networks, commonly known as Deep Learning (DL), are used to extract representation features from raw data through layered processing [43]. DL-based NIDSes usually convert packet payloads to sequences of bytes, characters or images. Recently, DL methods like Convolutional Neural Networks (CNNs) have been used to detect cyber-attacks [44]. CNNs can extract spatial and temporal traffic features from packet payloads without manual expert involvement.

Classical ML and one dimensional (1D) CNN were used to detect network attacks in [45]. While CNN was used for feature learning, attacks were detected using classical ML. In [46], images were generated from network traffic to detect malware using 2D CNNs. The images were created from raw traffic and passed to CNN which learned the traffic features. In [47], Marín et al. proposed *DeepMAL* to detect network traffic flows that contain malware. The authors used a DL model to

classify traffic using only raw byte streams without any handcrafted flow-based features. In a packet-based version of DeepMAL, the first 1024 bytes of payload were used to detect malware traffic. The authors suggested that payload-based DL approaches can outperform ML-based techniques in malware traffic detection. In contrast, Millar et al. [48] demonstrated that DL can detect malicious traffic with the first 50 bytes of payload. Furthermore, the authors suggested that classical ML can be more effective in classifying network traffic in certain situations.

2.3. Feature-fusion based intrusion detection

More recently, various combinations of traffic features for flow-based and payload-based detection approaches have been proposed [49,50]. Unlike methods that are purely flow-based or payload-based, feature-fusion based NIDSes use multiple types of network traffic to detect anomalies from different data sources. For example, *MFFAN* detected malicious traffic better than those that used flow-based features alone because it incorporated byte-level, packet-level and flow-level aspects of network traffic [11].

Similarly, an intrusion detection technique that used both statistical and payload-based features was proposed in TR-IDS [51]. Word embedding was used to map bytes of a packet payload to a word vector and Text-CNN was used for feature extraction [52]. Additionally, statistical features were extracted from network flows and packet headers. Both Text-CNN derived payload features and statistical features were concatenated and inputted to random forest for classification. In TR-IDS, payload features extracted from the first bytes of payload were used in conjunction with statistical traffic features. Generally, while CNNs and other DL methods can extract patterns from raw network data that might be challenging for classical ML algorithms, training DL models is computationally intensive and requires specialized hardware. Furthermore, DL models can be challenging to interpret and explain, raising concerns about transparency and trust.

In this paper, a multimodal ML-based NIDS is presented and evaluated. The proposed NIDS uses random forest [53], which is one of the well-known classical ML algorithms. TR-IDS [51] is the closest related work to the intrusion detection approach proposed in this paper. However, unlike TR-IDS [51], which concatenates payload and statistical features before classification, the combination of output in our NIDS occurs at the decision level. In the proposed multimodal NIDS, two separate random forest classification probabilities obtained using flow-based features and protocol payloads are combined using a soft voting scheme to make final class predictions.

Another important difference is that bigger size of packet payloads and larger number of flow-based features have been used in previous ML-based NIDSes including TR-IDS [51]. In contrast, only the first 32 bytes of protocol payload along with six flow-based traffic features are used in the proposed NIDS. Random forest is used as an ML classifier because it requires less computational resources as opposed to DL methods. The obtained results show that it is possible to detect most attacks using classical ML when both flow-based features and protocol payloads are used in a complementary manner.

3. The proposed multimodal ML-based NIDS

In this section, a multimodal Machine Learning (ML) based Network Intrusion Detection System (NIDS) that leverages both traffic flow features and payload data is presented. Decisions made by two ML classifiers trained using flow-based features and a small part of payload data, respectively, are used in a complementary manner to detect attacks. The proposed NIDS has flow-based and payload-based subsystems that take different aspects of network traffic as data objects for intrusion detection. The two types of network traffic enable the flow-based and payload-based subsystems to gather cyber threat intelligence from traffic flow features, and packet payloads, respectively. In both subsystems, random forest [53] is used as ML classifier. Finally, the classification results from the two random forest models are aggregated using a voting scheme.

Table 1
Random forest parameters.

Number of estimators	100
Split function	Gini
Maximum features for best split	2
Maximum tree depth	3
Minimum samples to split	2
Minimum leaf samples	0.1

Fig. 2 shows the traffic classification process of the proposed multimodal NIDS. Flow-based features and packet payload contents are extracted from raw traffic and two random forest classifiers are used to detect attacks. By considering distinct aspects of network traffic (flow-based features and payload contents), the proposed NIDS detects intrusions robustly. Firstly, flow-based features, which are handcrafted based on the domain knowledge of security professionals, are passed to random forest classifier as depicted in Fig. 2(a).

Secondly, the corresponding protocol payloads within a traffic flow are analyzed to scan for potential attack in user data. As shown in Fig. 2(b), protocol payloads from raw traffic are converted to fixed size numeric arrays at a byte-level. Note that packet switched networks fragment user data before transmission. Furthermore, some packets are used only to control the connection between hosts, and therefore, not all packets in a network flow contain user data or protocol payload. In addition, the size of protocol payload varies from one packet to the other. In the proposed NIDS, while packets with empty payload are discarded, non-empty payloads are zero-padded or trimmed to a fixed-size payload depending on the original size. Moreover, the hexadecimal strings of the fixed-size payloads are converted to numeric array before classification by the proposed NIDS.

As depicted in Fig. 2(a) and Fig. 2(c), a random forest model is trained on each type of network traffic. In both cases the data values are normalized before training the ML models. Whereas one random forest classifier is trained on flow-based features, decoded byte arrays of protocol payloads are used to train a second random forest classifier. Finally, the classification probabilities of the two ML models are used to obtain traffic class predictions using a soft voting scheme as depicted in Fig. 2(d).

3.1. Flow-based subsystem

The proposed multimodal NIDS detects intrusions in two ways. In the first part, flow-based features which are extracted from network traffic within a time window of interest are used to make one level of intrusion detection using a random forest classifier. In place of random forest, other supervised tree-based ML algorithms such as Gradient Boosting Decision Trees (GBDT) [54] and Extreme Gradient Boosting (XGBoost) [55] can be used as traffic classifiers to make the flow-based detection.

Randomized grid search was used to identify optimal hyperparameter values for the random forest classifier. This has been done by evaluating different parameter value combinations and cross validation to select best performing parameters. Table 1 shows the parameters of the classifier used in the flow-based subsystem.

Flow-based traffic features and protocol payloads are extracted from the same network flow. In other words, for a given traffic sample, the link between its protocol payloads and flow-based features is that both of them refer to the same network flow. A flow contains a group of packets having common traffic attributes and traversing through a network observation point within a designated time frame. Generally, packets in a flow are commonly aggregated based on *five-tuple* traffic features, namely source and destination Internet Protocol (IP) addresses, source and destination ports, and transport layer protocol.

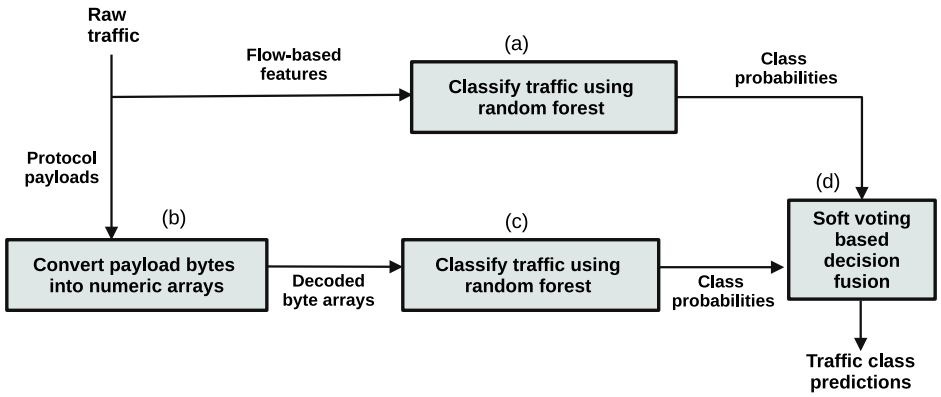

Fig. 2. Intrusion detection in the proposed multimodal ML-based NIDS.

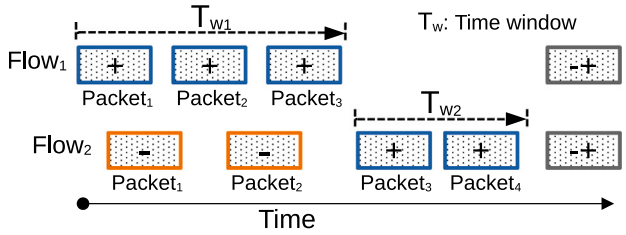

Fig. 3. Illustration of packets within flows and time windows (T_w). While blocks marked by "+" represent packets included in a current T_w, those with "-" marks show packets included in a previous T_w, and "-+" blocks represent packets waiting to be included in a future T_w of the respective flows.

Table 2
Flow-based features used in the multimodal NIDS.

Feature	Description
Flow starting time	Starting time of network flow
Flow duration	Time duration of network flow
Source bytes	Byte count from source
Destination bytes	Byte count from destination
Source packets	Packet count from source
Destination packets	Packet count from destination

In the proposed multimodal NIDS, flow-based traffic features and protocol payloads are extracted based on the concept of network flow. Assume *Flow* represents a set that contains all active network flows at an observation point within a time range of interest. *Flow* is a set defined by the notation in (1).

$$Flow = \{Flow_1, Flow_2, Flow_3, \ldots Flow_N\} \quad (1)$$

where N denotes the network flow count in a given time window.

In computer networks, exchange of control information and data occurs using packets. Here, $Flow_1$, $Flow_2$, …, $Flow_N$ denote a set of flows at an observation point where a total of N flows are processed. Within each flow, there are a number of packets. Fig. 3 shows individual network packets that form a *Flow* as $Packet_1$, $Packet_2$, $Packet_3$, …, $Packet_M$ where M represents the number of packets in the flow. A packet included in a flow may have user data in the form of protocol payload. While the objective of the flow-based subsystem is to identify elements of *Flow* that are suspected attack traffic by analyzing their statistical features, the payload-based subsystem analyzes actual contents of the packet payload for malicious data patterns.

In real world network scenarios, the elements of *Flow* change dynamically as the individual flows contained within it vary with time. Therefore, to account for the time-based variation in *Flow*'s contents, starting time and ending time associated with each element of *Flow* are defined. To illustrate with an example, consider the two *Flow* elements shown in Fig. 3. Each flow consists of IP packets where each one of them stays in the flow between specific starting and ending times.

Accordingly, without a loss of generality, a time window variable T_w is defined to represent a time range within which a set of packets in *Flow* are considered for intrusion detection. Depending on the time duration an intrusion detection is intended to cover, values of T_w can range from seconds to hours. As illustrated in Fig. 3, different T_w's cover a number of packets within *Flow* during the course of an intrusion detection. In this paper, T_w values ranging from 0 to 60 seconds were considered. The minimum and maximum values of T_w were determined by analyzing the training data. In the process of detecting attacks, it is operationally sensible to set a time range within which suspected intrusions are analyzed. Accordingly, the values of T_w were manually set in accordance with the minimum and maximum traffic flow duration of packets in the evaluation dataset.

For a traffic flow between source and destination, an ML classifier is trained using flow-based features. The objective of the flow-based subsystem is to detect attacks using limited flow information that can be obtained directly from packet headers with limited expert involvement. To this end, six flow-based features that are defined in the Internet Protocol Flow Information Export (IPFIX) standard have been identified [56]. These are byte counts and packet counts from source to destination and vice versa, flow starting time and duration. Since most network vendors have provisions for gathering and disseminating traffic flow data to support network management and security objectives, these statistical features are straightforward to collect with little expert involvement. To minimize bias in the flow-based random forest classifier, certain traffic attributes including source and destination IP addresses, source and destination ports, and protocol type were discarded, while the six standard traffic features were used for model training and testing. To extract the flow-based features from the UNSW-NB15 PCAP files, Argus [57] and Bro-IDS [58], also known as Zeek IDS, were used as network flow analyzers. From the flow-based features, some of those that are compliant to the IPFIX standard were used as flow-based features to train the propose multimodal NIDS. Table 2 shows a description of the flow-based features used in the proposed multimodal NIDS.

3.2. Payload-based subsystem

To complement results of the flow-based subsystem with a different source of cyber-threat intelligence, actual data contents of packets are used for intrusion detection in the payload-based subsystem. As shown in Fig. 2(c), pre-processed protocol payloads are passed to random forest for traffic classification. A motivation to use a mixed-data based intrusion detection comes from an observation about the diversity of network attacks. While some attacks (e.g. reconnaissance, port scanning) do not usually contain payload, other types of attacks (e.g. viruses, worms) come embedded within packet payloads.

Therefore, a complementary approach of attack detection, which is applicable in scenarios where at least some of the packets in a traffic flow contain payload, is proposed in this paper. By learning from flow-based traffic features on the one hand, and examining the first few bytes of protocol payload on the other, the proposed NIDS robustly detects diverse types network attacks.

Since the proposed NIDS depends on both flow-based traffic features and protocol payloads, it is critical to ensure that both types of data are from the same network flow. Therefore, in the payload-based subsystem, protocol payloads are taken from corresponding flows used in the flow-based subsystem. Accordingly, flow-specific payload data are extracted and labeled, which are in turn used in the payload-based subsystem to classify traffic. To this end, a new method shown in Algorithm 1 was developed and used to extract and label protocol payloads. Samples of flow-based features and protocol payloads are considered to belong to the same network flow if both have the same values for traffic features that are used as flow identifiers.

Algorithm 1 Protocol payload extractor in the multimodal NIDS.

Input:
Flow: Labeled dataset of network traffic flows
Raw: Unlabeled raw PCAP files
Proto: Transport layer protocol
Output: *Payload* and *Label*
function EXTRACT_PAYLOAD($Flow, Raw, Proto$)
  $Payload \leftarrow \emptyset$
  $Label \leftarrow \emptyset$
  while $Raw \neq \emptyset$ **do**
    $P_{flow} \leftarrow$ Extract *Proto*'s payload from *Raw* file
    $P_{id} \leftarrow$ Extract $[t_s, t_d, s_{ip}, d_{ip}, s_{port}, d_{port}]$ for P_{flow}
    $F_{id} \leftarrow$ Extract $[t_s, t_d, s_{ip}, d_{ip}, s_{port}, d_{port}]$ for *Flow*
    if $P_{flow} \neq \emptyset$ **then**
      if $P_{id} = F_{id}$ **then**
        $P_{label} \leftarrow F_{id}$'s label
        $Payload \leftarrow P_{flow} \cup Payload$
        $Label \leftarrow P_{label} \cup Label$
      else
        Continue
      end if
    end if
    return $Payload, Label$
  end while
end function

According to the definitions of *Flow* and T_w, *seven* traffic attributes are used as flow identifiers. Namely, flow starting time (t_s), time duration of the flow (t_d), source IP (s_{ip}), destination IP (d_{ip}), source port (s_{port}), destination port (d_{port}), and the transport layer protocol. Algorithm 1 is used to extract and label protocol payloads from raw packet capture (PCAP) files using labeled network flow dataset as a reference. Note that labeled flow-based traffic features are widely available as open source datasets. In contrast, labeled protocol payloads are not publicly available as far as we know. Therefore, the newly proposed procedure in Algorithm 1 is used to extract and label protocol payloads, which are in turn used by the payload-based subsystem to classify network traffic.

Algorithm 1 works as follows. Given a set of labeled network traffic flows and the corresponding unlabeled raw PCAP files, Algorithm 1 extracts and labels protocol payloads. The algorithm takes three input parameters, which are *Flow*, *Raw*, and *Proto*. *Flow* is a dataset that contains labeled network traffic flows while *Raw* consists of the original PCAP files of the dataset. *Proto* is the type of transport layer protocol whose payload is to be extracted and labeled. The output of Algorithm 1 is a list of protocol payloads and their corresponding labels.

The payload extraction and labeling procedure starts with empty lists, *Payload* and *Label*, which store the extracted payloads and their corresponding labels, respectively. As long as there are PCAP files to process, the algorithm extracts the payload of the specified transport layer protocol from *Raw*, and stores it in P_{flow}. To extract protocol payloads from *Raw*, Tshark [59], which is a well-known and open-source protocol analyzer, is used in Algorithm 1. From payloads in P_{flow}, a unique identifier (P_{id}) composed of timing parameters (t_s and t_d), s_{ip}, d_{ip}, s_{port}, and d_{port} is extracted from the raw packet data. A similar flow identifier (F_{id}) is extracted from the labeled IDS dataset.

If a non-empty payload is found, the algorithm checks whether the identifier of the extracted payload (P_{id}) matches the identifier of any flow in the labeled dataset (F_{id}). If there is a match (i.e., the extracted payload corresponds to a labeled flow in the dataset), the label of the matching flow in the labeled dataset is assigned to the extracted protocol payload ($P_{label} \leftarrow F_{id}$'s label). While the extracted payload (P_{flow}) is added to the list of payloads, the corresponding label (P_{label}) is included in the list of labels. If there is no match between the extracted payload and the flows in the labeled dataset, the algorithm continues to the next iteration until there are no more PCAP files. When there are no more PCAP files to process, the algorithm returns the lists of extracted payloads and their corresponding labels as the final output. In summary, Algorithm 1 iterates through the raw packet capture files, extracts protocol payloads of a specified transport layer protocol, and matches them with corresponding labeled flows in the dataset. The result is a list of extracted protocol payloads and their respective labels.

While the flow-based subsystem uses labeled traffic samples containing the six flow-based features as input, a list of protocol payloads extracted and labeled according to Algorithm 1 are used by the payload-based subsystem. As depicted in Fig. 2(c), a random forest classifier is trained to classify traffic based on these labeled protocol payloads. Flow-specific payloads are converted into numeric representations. Each byte in the payload's hexadecimal string is converted into its numeric representation.

Since the size of payload is variable across IP packets in a flow, a pre-processing procedure is applied to convert payload samples to fixed-size data. Algorithm 2 is used to transform the protocol payloads to byte-level traffic features that can be used for attack detection.

Algorithm 2 Payload pre-processor.

Input:
P: A list of protocol payloads in network flow
K: The first K number of bytes of payload
Output:
X: Numeric array representation of payloads
function DECODE_PAYLOAD(P, K)
  $X \leftarrow \emptyset$
  for i in P **do**
    $B \leftarrow$ Get the first K bytes of payload
    $M \leftarrow$ Numeric representation of B
    $N \leftarrow$ Normalized values of M
    $X \leftarrow X \cup N$
  end for
  return X
end function

For a protocol payload in a network flow, Algorithm 2 extracts the first set of bytes of the payload, converts them to numeric format and normalizes the values. The payload-based classification model then uses the normalized byte-level payload samples as training and testing data.

In effect, every byte of the payload is taken as a feature in the payload-based subsystem. While using the first few bytes of protocol payload for classification minimizes processing overhead, we also show in Section 4 that analyzing payload data beyond the first 64 bytes does not increase detection accuracy. Overall, a list of protocol payloads are taken, each payload's first few bytes are converted into numeric representation and the values are normalized. In this way, raw protocol payloads are converted into a numerical format suitable for random forest and other ML classifiers. Finally, a soft voting scheme is used to combine traffic class predictions from the flow-based and payload-based subsystems.

3.3. Classification using soft voting

In a voting based classification, predictions from individual ML classifiers are combined to obtain the final class prediction. Assume there are two ML classifiers denoted by h_1 and h_2. Each classifier predicts the probability that a given input sample, x, belongs to a class. In the proposed multimodal NIDS, h_1 and h_2 are the flow-based and payload-based subsystems. For a traffic sample x, let $P_j(x, h_1)$ and $P_j(x, h_2)$ denote the probability that x belongs to class j as determined by h_1 and h_2, respectively. In soft voting, the predicted probability for class j, denoted as $P_j(x)$, is the average of the predicted probabilities from the two subsystems as shown in Eq. (2).

$$P_j(x) = \frac{1}{2}\left(P_j(x, h_1) + P_j(x, h_2)\right) \tag{2}$$

$P_j(x, h_1)$ and $P_j(x, h_2)$ represent the predicted probabilities of class j for sample x according to classifiers h_1 and h_2, respectively.

For the given input sample x, the final class label is the class with the maximum average probability as shown in Eq. (3).

$$y_{\text{pred}}(x) = \arg\max_{j=1}^{C} P_j(x) \tag{3}$$

where $y_{\text{pred}}(x)$ is class label predicted for the input sample x, and C denotes the number of classes. By combining predicted traffic class probabilities of two independent random forest classifiers, the proposed multimodal ML-based NIDS leverages threat intelligence gathered from separate flow-based and payload-based subsystems and provides a robust intrusion detection.

4. Evaluation

In a learning phase, two random forest classifiers are trained on two distinct types of network traffic. While one classifier is trained on flow-based features, the second classifier uses protocol payloads as shown in Fig. 2. To detect intrusions during a testing phase, the flow-based and payload-based models are used to classify new samples of flow-based data and payload data, respectively. Subsequently, classification probabilities from each Machine Learning (ML) model are combined using a soft voting scheme to make the final traffic classification.

Since the proposed multimodal Network Intrusion Detection System (NIDS) uses both flow-based features and protocol payloads, two types of network traffic data are required for its evaluation. Accordingly, the proposed NIDS was evaluated using flow-based features and protocol payloads extracted from open source intrusion detection dataset known as *UNSW-NB15* [60]. The proposed method has been implemented using Scikit-learn and the Python programming language on Ubuntu operating system.

4.1. Evaluation dataset

The proposed multimodal NIDS was trained and tested using the UNSW-NB15 dataset [60]. While old datasets such KDDCup99 and NSL-KDD [61] were used in the past, neither of them forms a realistic representation of modern network traffic [62]. As the types of attacks have been increasing over time, it is crucial to include emerging attacks in an NIDS evaluation. Consequently, new datasets such as CDX [63], CICIDS2017 [64] and UNSW-NB15 [60] have been developed by researchers. After an evaluation of more than thirty different NIDS datasets, UNSW-NB15 is one of two datasets recommended by the authors in [65] due to its broad range of attack scenarios.

The UNSW-NB15 intrusion detection dataset was created by capturing nearly 100 gigabyte of Packet Capture (PCAP) files from an experimental computer network. It is one of the latest datasets and it contains traces of benign traffic and nine types of cyber-attacks. Network monitoring tools were used to extract 47 traffic features and to label network flows from the PCAP files. The labeled part of the dataset contains a total of 2540043 records in four files as Comma Separated Values (CSV). To evaluate the proposed multimodal NIDS, both the feature-ready CSV and the raw PCAP files were used.

Table 3
Feature importance.

Feature	Importance	Feature	Importance
swin	0.20379	sbytes	0.180505
spkts	0.170338	dwin	0.103253
isftplogin	0.076508	dpkts	0.07425
dbytes	0.070062	dmeansz	0.040409
dur	0.030362	synack	0.011084

Table 4
Flow-based TCP and UDP traffic samples in the dataset.

Protocol	Training samples (80%)	Testing samples (20%)	Total samples	Sample ratio by protocol
TCP	1196057	299014	1495071	58.86%
UDP	792347	198087	990434	39%
Others	–	–	54538	2.14%
Total			2540043	100%

As discussed in Section 3, six standard traffic features were extracted from data records in the original CSV files of the dataset. These flow-based features were selected due their availability in the IPFIX standard and ease of extraction from packet and byte counts of network transactions. The six features are among the statistical traffic information commonly collected by IPFIX compliant network devices. Furthermore, to assess the relative importance of the selected features, an ML model was trained on all flow-based features and then used to rank all features of the dataset including those used in the multimodal NIDS. Table 3 shows top contributing feature on a purely flow-based model trained using all traffic features. It can be observed that *spkts*, *sbytes*, *dbytes*, *dpkts* and *dur* are among the most important features in the flow-based model. Although there are other high ranking features in terms of the output of a standalone flow-based model, these have not been used in the proposed multimodal NIDS. Priority was given to the relative ease of feature extraction and IPFIX compliance of the selected features due to the use of both flow-based and payload-based features in a complementary manner in the proposed method.

In the flow-based part of the multimodal NIDS, traffic classification is made by random forest using the flow-based traffic features as shown in Fig. 2(a). Table 4 shows the network traffic samples of the dataset by protocol type. These flow-based traffic samples were obtained from the feature-ready CSV records of the dataset. As depicted in Table 4, the majority of the flows, 97.86%, are either TCP or UDP traffic. Accordingly, we evaluate the proposed multimodal NIDS using TCP and UDP traffic samples.

Similarly, protocol payloads were extracted from the original PCAP files using Algorithm 1 as discussed in Section 3. In the process of labeling protocol payloads, the CSV records were used as a reference to identify matching network flows. Following payload extraction and labeling, hexadecimal strings of the protocol payloads were converted to numeric values and normalized according to Algorithm 2. A second random forest classifier is then trained and tested using the labeled and normalized payload traffic samples.

TShark [59], which is a well known protocol analyzer, was used to extract network flows and protocol payloads from PCAP files. Particularly, starting timestamp, source and destination IPs, source and destination ports, and the respective TCP and UDP packet payloads were extracted from every PCAP file of the dataset. For both TCP and UDP, network flows without packet payloads were discarded. The same traffic features, except payload, were also extracted from the CSV files of the dataset. Redundant network flows were removed from the traffic features extracted using the PCAPs and CSV files. Subsequently, corresponding traffic samples from the CSV records and PCAPs were matched and selected according to the flow identifiers discussed in Section 3.

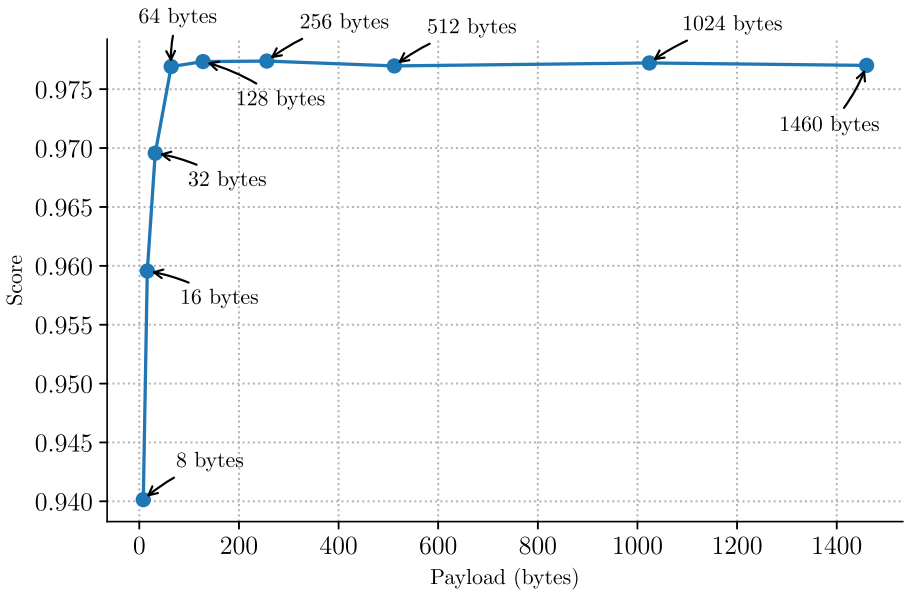

Fig. 4. Model F_1 score versus payload size.

Overall, more than ten million (10727720) TCP flows with payloads and more than seven hundred thousand (786118) UDP flows with payloads were extracted and labeled using Algorithm 1. Whereas all of the UDP flows were included, 10% of the TCP flows were randomly selected and used for evaluation. The resulting data was divided into 80% and 20% for training and testing, respectively.

4.2. Payload size

When protocol payloads are transformed to fixed-size samples according to Algorithm 2, the payload size was determined experimentally. While aiming for minimized resource consumption and processing time, it is also important to identify the size of protocol payload that gives better detection accuracy. To this end, the payload-based subsystem was trained and its detection performance evaluated across a range of payload sizes. Fig. 4 shows F_1 score of the payload-based model versus the number of bytes across different payload sizes. It can be seen that the model's F_1 score reaches 97% at a payload size of 32 bytes. Furthermore, while processing the payload data beyond the first 64 bytes incurs additional computational costs, no significant performance improvement is obtained by doing so.

Similarly, different fixed-size payloads starting from the leading end of the payload have been used in other payload-based detection approaches [46–48]. Although the exact payload size varies from one method to the other, it is common to take traffic samples from the first few bytes of the payload. Cognizant of resources and time required to handle protocol payloads of 64 bytes and 32 bytes vis-a-vis the detection accuracy achieved by using each, the first 32 bytes of payload were considered in the proposed method. Accordingly, the payload-based model was trained and tested using the first 32 bytes of protocol payload.

With the aforementioned configurations of the flow-based and payload-based models, the multimodal NIDS combines classifications outputs of the two models using the soft voting scheme discussed in Section 3.3. When the models were trained on their respective data samples, they could learn parameters of the data too well and result in overfitting, in which case the models would not perform well on new data. Therefore, to tackle overfitting, the models were trained using a *k-fold* cross-validation method [66,67]. Each model was trained and validated on the available data at different rounds of training by using *ten-fold* cross validation ($k = 10$), where one out of ten traffic samples were used for validation and the rest for training. Accordingly, the proposed model was evaluated by taking average scores of evaluation metrics discussed in Section 4.3.

4.3. Evaluation metrics

The correctness of the predictions made by the proposed multimodal NIDS was measured using a confusion matrix. Confusion matrix was used to compare traffic classes predicted by the proposed NIDS against the actual traffic classes (ground truth). An ideal NIDS would have non-zero values for True Positive (TP) and True Negative (TN), and zero values for False Positive (FP) and False Negative (FN) which represent errors.

The objective of the proposed NIDS is to classify network traffic into attack or normal, and into multiple attack categories in the case of multiclass classification. The confusion matrix was used in the evaluation of binary as well as multiclass traffic classification. In the case of the multiclass classification, where the NIDS predicted not only normal and attack traffic classes but also the types of attack, the confusion matrix was used to provide a detailed breakdown of the predictions for more than two traffic types. Accordingly, the classification performance of the proposed multimodal NIDS in predicting normal traffic and categories of attack traffic was evaluated using average *Accuracy*, *Precision*, *Recall*, F_1 score, and *False Positive Rate (FPR)*. These standard metrics are calculated from TP, TN, FP, and FN entries of the confusion matrix for the respective traffic class [68].

Furthermore, *Receiver Operating Characteristic (ROC)* curve, which shows True Positive Rate (TPR) and FPR, was used for evaluation. The target of the NIDS is maximizing TPR while minimizing FPR. In the case of multiclass classification, ROC curves with *One-vs-Rest (OvR)* approach were used to measure the correctness of predictions. Using OvR approach, the classification performance of the NIDS was evaluated for each traffic class. In OvR, the traffic class under consideration is treated as the positive class while considering all other traffic classes as the negative class. In addition, *Area Under the Curve (AUC)* measures the classification performance by calculating the area under the ROC curve. A nearly perfect NIDS would classify almost all network traffic correctly, and the AUC in that case would be very close to 1.

Flow-based features and protocol payloads from TCP and UDP of the UNSW-NB15 dataset were used for training and testing. The reason that the proposed NIDS was evaluated using only these types of traffic is because TCP and UDP are the most commonly used transport layer protocols. In addition, 97.86% of all data samples in the UNSW-NB15

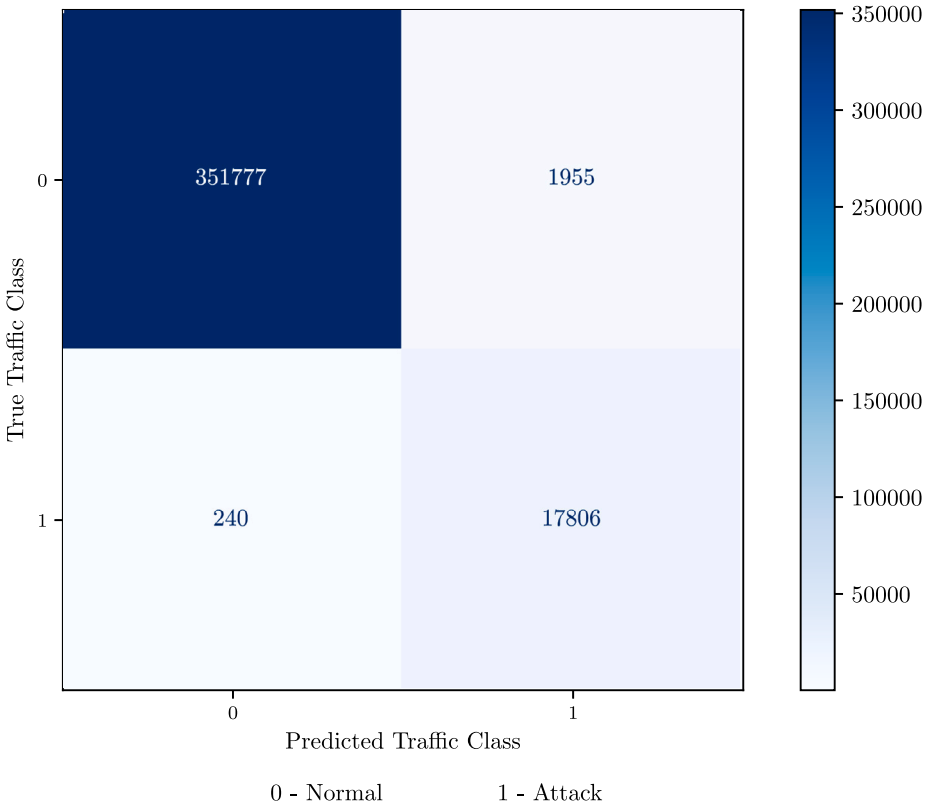

Fig. 5. Binary confusion matrix of the proposed multimodal NIDS.

dataset come from either TCP or UDP traffic flows. Therefore, the proposed NIDS was evaluated using these traffic samples. The proposed NIDS was tested for binary and multiclass classification.

5. Results

In this section, the evaluation results of the proposed method are presented. Fig. 5 shows the binary confusion matrix of the proposed multimodal NIDS using traffic samples in the test data. While *True Traffic Class* shows the number of samples of attack and normal traffic in reality (ground truth), values of *Predicted Traffic Class* are the respective numbers of attack and normal traffic as classified by the proposed NIDS. As shown in Fig. 5, out of a total of 371,778 testing samples, 369,583 were correctly classified by the model. The remaining 2195 traffic samples were misclassified where 1955 were false positives and 240 were false negatives. For binary classification, the proposed model was evaluated using average values of Accuracy, Precision, Recall and F_1 metrics using k-fold cross validation scores. The model achieved more than 98% on most of the evaluation metrics. The obtained high scores on the evaluation metrics indicate that the proposed NIDS can classify normal and attack traffic accurately. Fig. 6 shows the ROC curves of evaluating the proposed NIDS for multiclass classification. As stated earlier in this section, a good classifier has a ROC curve that is close to the upper left corner of the plot, which indicates high true positive rates and low false positive rates. As can be seen in Fig. 6, the ROC curves of the attack traffic except two are close to the upper left corner of the plot. Backdoor and Worms are the two attack traffic classes for which the proposed NIDS scored relatively low.

Another way to assess the effectiveness of the proposed NIDS for multiclass classification is to look at the AUC of the ROC curve. Values of AUC close to 1 indicate a good classification performance. As shown in Fig. 6, the ROC curves of most traffic classes have high AUCs except Backdoor and Worms, which have AUCs of 0.85 and 0.94, respectively. Furthermore, very low FPRs were obtained for the nine attack classes. In practice, excessive false alerts tend to cause threat-alert fatigue and take the attention of a security expert from responding to real attacks. Therefore, the fact that the proposed multimodal NIDS has low FPR has useful practical significance for network security operations.

Overall, the model classified most traffic classes correctly with AUCs of more than 0.95 in most cases. Exceptions to this are Worms and Backdoor attack types with AUCs of 0.85 and 0.94, respectively. The obtained results confirm that the proposed multimodal NIDS can classify most normal and attack traffic correctly. However, the classification results for attacks such as Backdoor and Worms were not as good as the others. In these cases, the low scores could be due to the fact that the attacks are a minority in the dataset with relatively few samples in the training and testing sets. The number of samples of each type of attack used for evaluation are shown in multiclass confusion matrix in Fig. 7.

Fig. 7 shows multiclass confusion matrix for the proposed model. While values to the right of a *True Traffic Class* label are the real number of samples of that class, values shown vertical to *Predicted Traffic Class* label are the number of samples predicted by the model as the respective class. It can be seen that the cells along the diagonal of the confusion matrix from top left to bottom right, contain most of the non-zero values, indicating the proposed NIDS detects most attacks correctly. In comparison to the actual attack categories, it can be seen that the model has predicted most traffic samples correctly. However, classes such as Normal, Exploits, Generic and Fuzzers had a few of their samples misclassified as other types of traffic.

In summary, the proposed multimodal NIDS, which uses flow-based features and the first 32 bytes of protocol payload in a complementary manner was able to classify network traffic successfully, with high average Accuracy, Precision, Recall, F_1 score and low FPR values. While the proposed NIDS identified most attack categories in the evaluation dataset accurately, it has a limitation in terms of detecting unknown attacks. Since random forest, which is a supervised ML, is used in both the flow-based and payload-based models of the proposed multimodal NIDS, it does not detect unknown or zero-day cyber-attacks.

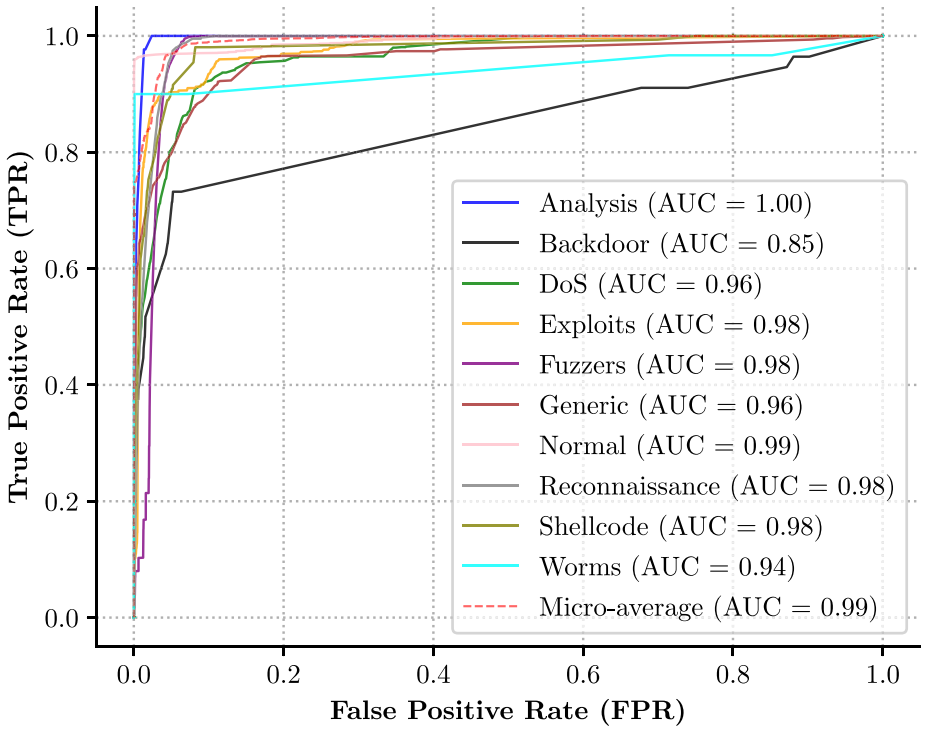

Fig. 6. Model ROC curves and AUCs.

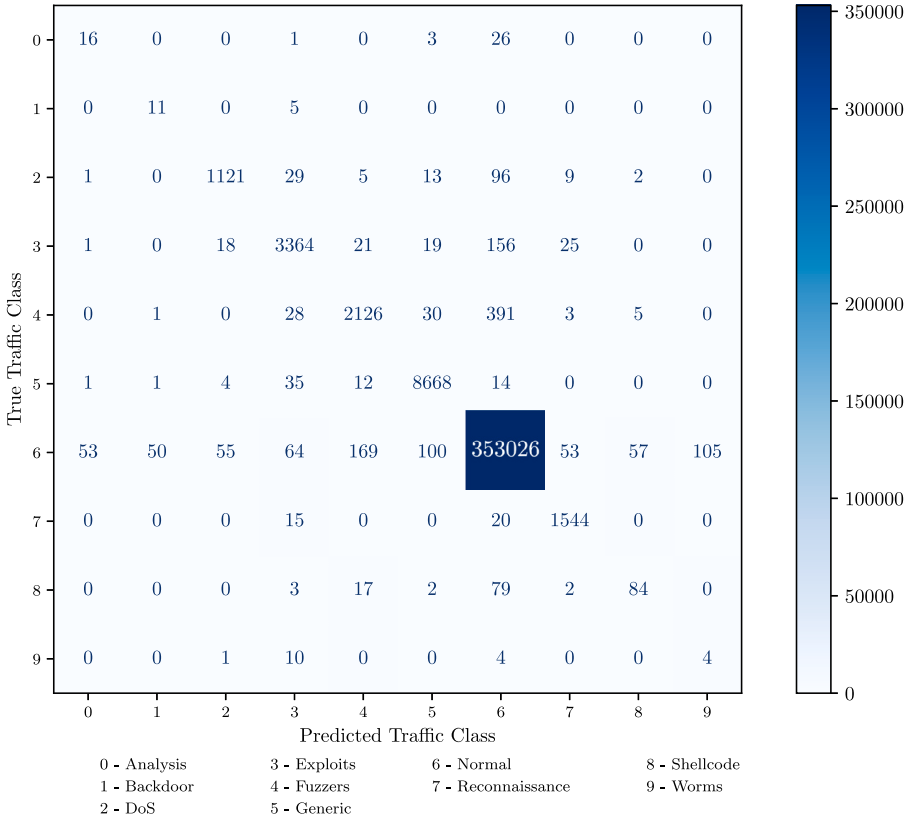

Fig. 7. Multiclass confusion matrix.

5.1. Model explainability

With the widespread use of artificial intelligence (AI) in many domains, explainable AI (XAI) [69,70] has been used to interpret results of intrusion detection models [71]. Beyond detecting attacks accurately, it is important that the decisions made by an ML-based NIDS are transparent and explainable. Once malicious network flows have been identified by the NIDS, model explainability helps to determine

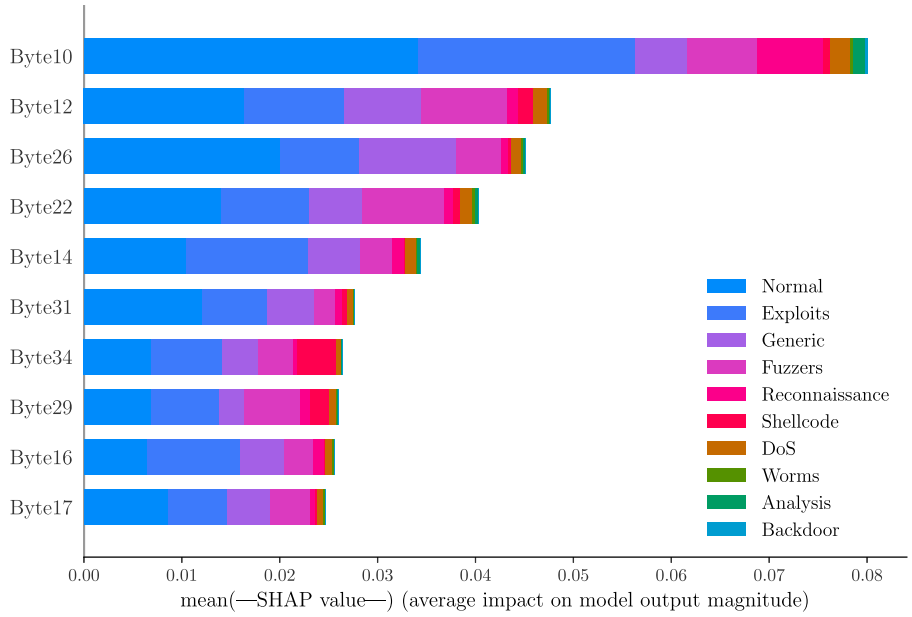

Fig. 8. SHAP explainability of multiclass model output.

why the attacks occurred and to inform actions that can be taken to deal with the attacks. Model explainability provides insights about the traffic attributes that the model relies on to detect attacks.

To gain insights about the explainability of classification outputs of the proposed multimodal NIDS, Shapley Additive Explanations (SHAP) values were used [72]. Model explainability was analyzed using traffic features used in the payload-based and flow-based models that form the proposed multimodal NIDS. Fig. 8 shows summary SHAP plot of the payload-based model for multiclass classification. Fig. 8 depicts the impact of different payload byte features on the prediction of normal traffic and nine types of attacks. The results indicate that the 10th byte of the payload is the highest contributing feature to the model's output. In addition, while the individual contributions to the model output vary on different payload byte positions, most top contributor features have been found to be up to the 32nd byte of the payload. A similar trend was observed in the case of binary classification of the model. The results indicate that the treatment of the protocol payload at the byte-level leads to the identification of the most common payload byte position that can be used for attack detection.

Similarly, the contribution of flow-based features to the multimodal output was evaluated using SHAP values of the flow-based model. The SHAP values of the flow-based features were smaller than that of the payload-based model, thus indicating flow-based features have smaller contribution to the overall multimodal model output. Although the SHAP values of the flow-based features were small, their presence in the multimodal NIDS was necessary for the detection of attacks that have limited payload.

5.2. Comparisons

In this subsection, the proposed multimodal NIDS is compared with three types of intrusion detection approaches. Firstly, the proposed multimodal NIDS is compared with standalone flow-based and payload-based detection methods. Secondly, the decision-level combination of results from the two models is compared against a method that uses aggregated flow-based and payload-based features at the data level. The fusion of classification results in the proposed method occurs at the decision level using a soft voting scheme. However, in multimodal ML approaches, the fusion can also be made at an intermediate step or early at a feature level.

Due to the use of random forest as a classifier, intermediate fusion is not applicable in the proposed method. Therefore, thirdly, the proposed decision-level multimodal NIDS is compared with a detection approach that combines flow-based data and the corresponding protocol payloads as features before classification. Accordingly, a random forest classifier was trained on a combined data obtained by joining the flow-based and payload-based traffic samples. The flow-based features were zero-padded to match the byte-level features of the payload data.

In all three cases, random forest was used as the classifier to make a fair comparison with the proposed method. The results show that the proposed multimodal NIDS is more effective in detecting attacks than both standalone flow-based and payload-based approaches. While the payload-based model scored better than the flow-based model in almost all evaluation metrics, the proposed multimodal model scored better than both of them.

To assess the effectiveness of the decision-level fusion of results in the proposed method, it was compared with an early fusion approach. Fig. 9 shows ROC curves of the model that was trained using traffic samples combined at a feature-level. In contrast, as shown in Fig. 6, the ROC curves of the proposed method have higher AUC scores. The results indicate that early fusion of flow-based features and protocol payloads has lower attack detection performance than the proposed multimodal model which uses decision-level fusion of results.

To compare the different fusion approaches in terms of explainability, Shapley Additive Explanations (SHAP) values were used to evaluate the contributions of the features on the output of the models. While it was possible to measure the impact of payload and flow-features using SHAP values as shown in Section 5.1, the early fusion based model showed limitations in terms of explainability. For the model based on the fusion of flow and payload features at data-level, model output was not directly attributable to specific flow or payload features. The early fusion of flow and payload data before classification introduces complexity in the explainability of model output.

6. Conclusions

In this paper, a new multimodal Machine Learning (ML) based Network Intrusion Detection System (NIDS) is presented. The multimodal NIDS uses distinct aspects of traffic in a complementary approach to reliably detect network intrusions. It has flow-based and payload-based subsystems, where six standard traffic flow features and the first 32

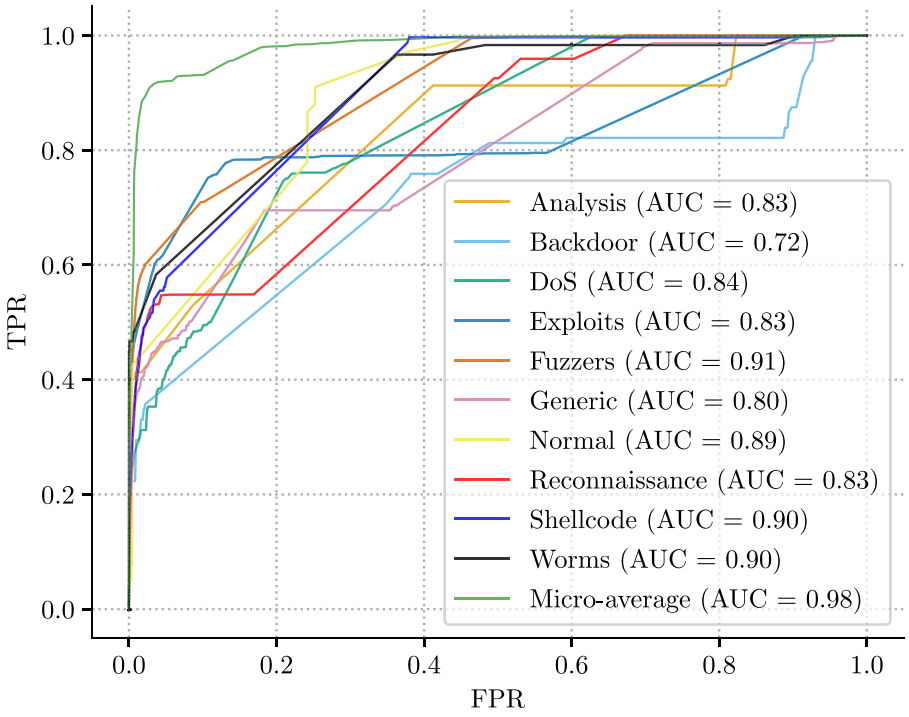

Fig. 9. ROC curves using early fusion.

bytes of protocol payload are processed, respectively. In both subsystems, a random forest ML model is trained to classify network traffic. Predicted probabilities of the two subsystems are then aggregated using a soft voting scheme to obtain the final classification results. The proposed NIDS was evaluated for binary and multiclass classification using flow-based features and protocol payloads extracted from a publicly available UNSW-NB15 dataset.

The results show that the multimodal NIDS can detect network attacks with high average Precision, Recall, F_1, and Accuracy scores reaching 98% in many cases, and up to 99% in some cases. The FPR scores of the proposed NIDS were low for most of the attack categories in the evaluation dataset. While the proposed NIDS uses six standardized flow-based features and the first 32 bytes of protocol payload, most previous ML-based NIDS used large number of traffic features and big payload sizes. Therefore, the results in this paper confirm that a multimodal ML-based NIDS that learns from few standard traffic flow features on the one hand, and checks the first few bytes of protocol payload on the other, has a good potential to detect most network attacks. For future work, the applicability and effectiveness of the proposed NIDS for online intrusion detection will be evaluated. The implementation of the multimodal ML-based NIDS is publicly available on GitHub.[1]

CRediT authorship contribution statement

Aklil Kiflay: Conceptualization, Investigation, Methodology, Software, Visualization, Writing – original draft. **Athanasios Tsokanos:** Conceptualization, Funding acquisition, Project administration, Resources, Supervision, Validation, Writing – review & editing. **Mahmood Fazlali:** Conceptualization, Funding acquisition, Project administration, Resources, Supervision, Validation, Writing – review & editing. **Raimund Kirner:** Conceptualization, Funding acquisition, Project administration, Resources, Supervision, Validation, Writing – review & editing.

[1] https://github.com/azkiflay/multimodal-nids

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Link to code is provided in the paper and the dataset used is open source and available publicly.

Acknowledgments

This work was supported by the School of Physics, Engineering and Computer Science, University of Hertfordshire, Hatfield, United Kingdom.

References

[1] Abdulganiyu OH, Ait Tchakoucht T, Saheed YK. A systematic literature review for network intrusion detection system (IDS). Int J Inf Secur 2023;1–38.
[2] Chapaneri R, Shah S. A comprehensive survey of machine learning-based network intrusion detection. Smart Intell Comput Appl 2019;345–56.
[3] Shaukat K, Luo S, Varadharajan V, Hameed IA, Xu M. A survey on machine learning techniques for cyber security in the last decade. IEEE Access 2020;8:222310–54.
[4] Xin Y, Kong L, Liu Z, Chen Y, Li Y, Zhu H, et al. Machine learning and deep learning methods for cybersecurity. Ieee Access 2018;6:35365–81.
[5] Torres JM, Comesaña CI, Garcia-Nieto PJ. Machine learning techniques applied to cybersecurity. Int J Mach Learn Cybern 2019;10(10):2823–36.
[6] Pacheco F, Exposito E, Gineste M, Baudoin C, Aguilar J. Towards the deployment of machine learning solutions in network traffic classification: A systematic survey. IEEE Commun Surv Tutor 2018;21(2):1988–2014.
[7] Buczak AL, Guven E. A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Commun Surv Tutorials 2015;18(2):1153–76.
[8] Fazlali M, Khodamoradi P, Mardukhi F, Nosrati M, Dehshibi MM. Metamorphic malware detection using opcode frequency rate and decision tree. Int J Inf Secur Priv (IJISP) 2016;10(3):67–86.
[9] Aslan ÖA, Samet R. A comprehensive review on malware detection approaches. IEEE Access 2020;8:6249–71.

[10] Khraisat A, Gondal I, Vamplew P, Kamruzzaman J. Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity 2019;2(1):1–22.

[11] Huang W, Han X, Zhang M, Li M, Liu W, Yang Z, et al. MFFAN: Multiple features fusion with attention networks for malicious traffic detection. In: 2022 IEEE international conference on trust, security and privacy in computing and communications (trustCom). IEEE; 2022, p. 391–8.

[12] Thakkar A, Lohiya R. A review on challenges and future research directions for machine learning-based intrusion detection system. Arch Comput Methods Eng 2023;1–25.

[13] de Melo LH, de C Bertoli G, Pereira LA, Saotome O, Domingues MF, dos Santos AL. Generalizing flow classification for distributed denial-of-service over different networks. In: GLOBECoM 2022-2022 IEEE global communications conference. IEEE; 2022, p. 879–84.

[14] Apruzzese G, Colajanni M. Evading botnet detectors based on flows and random forest with adversarial samples. In: 2018 IEEE 17th international symposium on network computing and applications. NCA, IEEE; 2018, p. 1–8.

[15] Umer MF, Sher M, Bi Y. Flow-based intrusion detection: Techniques and challenges. Comput Secur 2017;70:238–54.

[16] Özdel S, Ateş Ç, Ateş PD, Koca M, Anarım E. Payload-based network traffic analysis for application classification and intrusion detection. In: 2022 30th European signal processing conference. EUSIPCO, IEEE; 2022, p. 638–42.

[17] Kiflay AZ, Tsokanos A, Kirner R. A network intrusion detection system using ensemble machine learning. In: 2021 international carnahan conference on security technology. ICCST, IEEE; 2021, p. 1–6.

[18] Al-Bakaa A, Al-Musawi B. Flow-based intrusion detection systems: A survey. In: International conference on applications and techniques in information security. Springer; 2021, p. 121–37.

[19] Nguyen LG, Watabe K. Flow-based network intrusion detection based on BERT masked language model. In: Proceedings of the 3rd international CoNEXT student workshop. 2022, p. 7–8.

[20] Alasmary F, Alraddadi S, Al-Ahmadi S, Al-Muhtadi J. Shieldrnn: A distributed flow-based ddos detection solution for iot using sequence majority voting. IEEE Access 2022;10:88263–75.

[21] Thakkar A, Lohiya R. Fusion of statistical importance for feature selection in deep neural network-based intrusion detection system. Inf Fusion 2023;90:353–63.

[22] Santos L, Gonçalves R, Rabadao C, Martins J. A flow-based intrusion detection framework for internet of things networks. Cluster Comput 2021;1–21.

[23] Zhou Y, Cheng G, Jiang S, Dai M. Building an efficient intrusion detection system based on feature selection and ensemble classifier. Comput Netw 2020;174:107247.

[24] Kshirsagar D, Kumar S. Towards an intrusion detection system for detecting web attacks based on an ensemble of filter feature selection techniques. Cyber-Phys Syst 2023;9(3):244–59.

[25] Mishra P, Varadharajan V, Tupakula U, Pilli ES. A detailed investigation and analysis of using machine learning techniques for intrusion detection. IEEE Commun Surv Tutorials 2018;21(1):686–728.

[26] Di Mauro M, Galatro G, Fortino G, Liotta A. Supervised feature selection techniques in network intrusion detection: A critical review. Eng Appl Artif Intell 2021;101:104216.

[27] Li J, Cheng K, Wang S, Morstatter F, Trevino RP, Tang J, et al. Feature selection: A data perspective. ACM Comput Surv (CSUR) 2017;50(6):1–45.

[28] Bommert A, Sun X, Bischl B, Rahnenführer J, Lang M. Benchmark for filter methods for feature selection in high-dimensional classification data. Comput Statist Data Anal 2020;143:106839.

[29] Pontes CF, De Souza MM, Gondim JJ, Bishop M, Marotta MA. A new method for flow-based network intrusion detection using the inverse Potts model. IEEE Trans Netw Serv Manag 2021;18(2):1125–36.

[30] Wang F, Chai G, Li Q, Wang C. An efficient unsupervised domain adaptation deep learning model for unknown malware detection. In: Security and privacy in new computing environments: 4th eAI international conference, SPNCE 2021, virtual event, December 10-11, 2021, proceedings. Springer; 2022, p. 64–76.

[31] Siddique K, Akhtar Z, Khan FA, Kim Y. KDD cup 99 data sets: A perspective on the role of data sets in network intrusion detection research. Computer 2019;52(2):41–51.

[32] Sarhan M, Layeghy S, Moustafa N, Portmann M. Netflow datasets for machine learning-based network intrusion detection systems. In: Big data technologies and applications: 10th EAI international conference, BDTA 2020, and 13th EAI international conference on wireless internet, WiCON 2020, virtual event, December 11, 2020, proceedings 10. Springer; 2021, p. 117–35.

[33] Apruzzese G, Pajola L, Conti M. The cross-evaluation of machine learning-based network intrusion detection systems. IEEE Trans Netw Serv Manag 2022;19(4):5152–69.

[34] Sperotto A, Schaffrath G, Sadre R, Morariu C, Pras A, Stiller B. An overview of IP flow-based intrusion detection. IEEE Commun Surv Tutorials 2010;12(3):343–56.

[35] Soltani M, Siavoshani MJ, Jahangir AH. A content-based deep intrusion detection system. Int J Inf Secur 2022;1–16.

[36] Cavnar WB, Trenkle JM, et al. N-gram-based text categorization. In: Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval. Vol. 161175, Las Vegas, NV; 1994, p. 14.

[37] Wressnegger C, Schwenk G, Arp D, Rieck K. A close look on n-grams in intrusion detection: anomaly detection vs. classification. In: Proceedings of the 2013 ACM workshop on artificial intelligence and security. 2013, p. 67–76.

[38] Wang K, Parekh JJ, Stolfo SJ. Anagram: A content anomaly detector resistant to mimicry attack. In: International workshop on recent advances in intrusion detection. Springer; 2006, p. 226–48.

[39] Swarnkar M, Hubballi N. Rangegram: A novel payload based anomaly detection technique against web traffic. In: 2015 IEEE international conference on advanced networks and telecommunications systems. ANTS, IEEE; 2015, p. 1–6.

[40] Wang K, Stolfo SJ. Anomalous payload-based network intrusion detection. In: International workshop on recent advances in intrusion detection. Springer; 2004, p. 203–22.

[41] Perdisci R, Ariu D, Fogla P, Giacinto G, Lee W. McPAD: A multiple classifier system for accurate payload-based anomaly detection. Comput Netw 2009;53(6):864–81.

[42] Swarnkar M, Hubballi N. OCPAD: One class naive Bayes classifier for payload based anomaly detection. Expert Syst Appl 2016;64:330–9.

[43] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature 2015;521(7553):436–44.

[44] Tsogbaatar E, Bhuyan MH, Fall D, Taenaka Y, Gonchigsumlaa K, Elmroth E, et al. A 1D-CNN based deep learning for detecting VSI-ddos attacks in IoT applications. In: Advances and trends in artificial intelligence. artificial intelligence practices: 34th international conference on industrial, engineering and other applications of applied intelligent systems, IEA/AIE 2021, Kuala Lumpur, Malaysia, July 26–29, 2021, proceedings, part I 34. Springer; 2021, p. 530–43.

[45] Xu Y, Zhang X, Ye T, Qiu Z, Zhang L, Zhang H, et al. 1D cnn for feature reconstruction on network threat detection. In: 2021 13th international conference on machine learning and computing. 2021, p. 127–32.

[46] Wang W, Zhu M, Zeng X, Ye X, Sheng Y. Malware traffic classification using convolutional neural network for representation learning. In: 2017 international conference on information networking. ICOIN, IEEE; 2017, p. 712–7.

[47] Marín G, Caasas P, Capdehourat G. Deepmal-deep learning models for malware traffic detection and classification. In: Data science–analytics and applications: proceedings of the 3rd international data science conference–IDSC2020. Springer; 2021, p. 105–12.

[48] Millar K, Cheng A, Chew HG, Lim C-C. Deep learning for classifying malicious network traffic. In: Trends and applications in knowledge discovery and data mining: PAKDD 2018 workshops, BDASC, BDM, ML4Cyber, PAISI, DaMEMO, Melbourne, VIC, Australia, June 3, 2018, revised selected papers 22. Springer; 2018, p. 156–61.

[49] Lin K, Xu X, Xiao F. MFFusion: A multi-level features fusion model for malicious traffic detection based on deep learning. Comput Netw 2022;202:108658.

[50] Lin Y-D, Wang Z-Y, Lin P-C, Nguyen V-L, Hwang R-H, Lai Y-C. Multi-datasource machine learning in intrusion detection: Packet flows, system logs and host statistics. J Inf Secur Appl 2022;68:103248.

[51] Min E, Long J, Liu Q, Cui J, Chen W. TR-IDS: Anomaly-based intrusion detection through text-convolutional neural network and random forest. Secur Commun Netw 2018;2018.

[52] Kim Y. Convolutional neural networks for sentence classification. 2014, arXiv preprint arXiv:1408.5882.

[53] Breiman L. Random forests. Mach Learn 2001;45(1):5–32.

[54] Friedman JH. Greedy function approximation: a gradient boosting machine. Ann Stat 2001;1189–232.

[55] Chen T, Guestrin C. Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 2016, p. 785–94.

[56] Claise B, Trammell B, Aitken P. Specification of the IP flow information export (IPFIX) protocol for the exchange of flow information. Tech. rep., 2013.

[57] Bullard C. Argus, Online: https://openargus.org/, [Accessed 13/08/2022].

[58] Zeek IDS, Online: https://zeek.org/, [Accessed 20/08/2022].

[59] Combs G. Tshark, Online: http://www.wireshark.org/docs/man-pages/tshark.html, [Accessed 03/07/2022].

[60] Moustafa N, Slay J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: 2015 military communications and information systems conference (MilCIS). IEEE; 2015, p. 1–6.

[61] Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the KDD CUP 99 data set. In: 2009 IEEE symposium on computational intelligence for security and defense applications. Ieee; 2009, p. 1–6.

[62] Sommer R, Paxson V. Outside the closed world: On using machine learning for network intrusion detection. In: 2010 IEEE symposium on security and privacy. IEEE; 2010, p. 305–16.

[63] Sangster B, O'connor T, Cook T, Fanelli R, Dean E, Morrell C, et al. Toward instrumenting network warfare competitions to generate labeled datasets. In: CSET. 2009.

[64] Sharafaldin I, Lashkari AH, Ghorbani AA. Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp 2018;1:108–16.

[65] Ring M, Wunderlich S, Scheuring D, Landes D, Hotho A. A survey of network-based intrusion detection data sets. Comput Secur 2019;86:147–67.

[66] Wong T-T, Yeh P-Y. Reliable accuracy estimates from k-fold cross validation. IEEE Trans Knowl Data Eng 2019;32(8):1586–94.

[67] Wong T-T. Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation. Pattern Recognit 2015;48(9):2839–46.
[68] Sokolova M, Lapalme G. A systematic analysis of performance measures for classification tasks. Inf Process Manag 2009;45(4):427–37.
[69] Rjoub G, Bentahar J, Wahab OA, Mizouni R, Song A, Cohen R, et al. A survey on explainable artificial intelligence for cybersecurity. IEEE Trans Netw Serv Manag 2023.
[70] Minh D, Wang HX, Li YF, Nguyen TN. Explainable artificial intelligence: a comprehensive review. Artif Intell Rev 2022;1–66.
[71] Wang M, Zheng K, Yang Y, Wang X. An explainable machine learning framework for intrusion detection systems. IEEE Access 2020;8:73127–41.
[72] Lundberg SM, Erion G, Chen H, DeGrave A, Prutkin JM, Nair B, et al. From local explanations to global understanding with explainable AI for trees. Nat Mach Intell 2020;2(1):56–67.