On Explainable and Adaptable Detection of Distributed Denial-of-Service Traffic

Yebo Feng, Jun Li, *Senior Member, IEEE*, Devkishen Sisodia, and Peter Reiher, *Member, IEEE*

***Abstract*—Launched from numerous end-hosts throughout the Internet, a distributed denial-of-service (DDoS) attack can exhaust the network bandwidth or other resources of a victim, cripple its service, and make it unavailable to legitimate clients. Recently many learning-based approaches attempt to detect DDoS attacks, but their results are often hardly explainable to users and their models are seldom adaptable to new environments. In this paper, we propose a new learning-based DDoS detection approach. It detects DDoS attacks via an enhanced k-nearest neighbors (KNN) algorithm, which utilizes a k-dimensional (KD) tree to speed up the detection process, and classifies DDoS sources at a fine granularity according to each IP's risk level. Compared to previous DDoS detection approaches, this approach outputs explanatory information that enables network administrators to easily inspect detection results and make necessary interventions. Moreover, this approach is adaptable in that users do not need to retrain the detection model to have it fit with a new network environment. We evaluated this approach in both simulated environments and the real world, achieving more than 95.6% accuracy in detecting DDoS attacks at line speed. In addition, we carried out a human subject study on its explainability, demonstrating that the outputs can help people better understand the attack and make interventions precisely and promptly.**

***Index Terms*—Anomaly detection, DDoS detection, distributed denial -of- service (DDoS), explainable machine learning, K-nearest neighbors (KNN), principal component analysis (PCA), traffic analysis.**

I. Introduction

Distributed denial-of-service (DDoS) attacks pose a severe security problem on today's Internet and can render servers, network infrastructure, and applications unavailable to their users. They overwhelm the targeted machine or network resources with excessive traffic, thereby preventing legitimate traffic from being processed [1]. Cisco indicated in their March 2020 white paper [2] that the frequency of DDoS attacks had increased more than 2.5 times and the average size of DDoS attacks had approached 1 Tbps over the last three years.

Of foremost importance in DDoS defense tasks are to detect DDoS, classify DDoS sources, and do so accurately and quickly. Decades of research and industry efforts have led to a myriad of DDoS detection and classification approaches. In recent years, many researchers have begun to harness machine learning algorithms, such as support vector machine (SVM), Naive Bayes, convolutional neural network (CNN), etc., on Big Data in detecting and classifying DDoS attacks (e.g., [3], [4], [5]). The evaluations of such approaches demonstrate their strong ability in extracting useful knowledge from massive training data and decent recall scores in detecting a variety of DDoS attacks.

Unfortunately, the negative aspects of most learning-based approaches are also apparent. First, many learning-based approaches may not be well-suited for practical applications, as their detection results are often difficult to interpret, resembling black boxes [6], [7]. As a result, extracting explanatory information from the detection outputs generated by these methods (e.g., deep neural networks and deep recurrent neural networks) is challenging. In real-world deployments, network administrators particularly need good explainability, as they usually have to manually review and verify DDoS detection results, including eliminating false alarms and avoiding severe collateral damage due to filtering traffic from legitimate users. This is especially true for large-scale networks, such as Internet service providers (ISPs) and Internet exchange points (IXPs), where a single filtering rule can disconnect a considerable number of IP addresses, making network administrators hesitant about which actions to take. According to previous literature [8], [9], [10] and our analysis (Section III), detection approaches with useful explanatory information should possess three features to help network administrators make appropriate and timely decisions:

- *Transparent:* The detection model should allow users to gain clear insight into the traffic processing procedures, intuitively illustrating all network contexts and situations.
- *Traceable:* The detection outputs should help users quickly understand the detection logic and indicate root causes.
- *Heuristic:* The detection outputs should help users make applicable decisions to address the ongoing attack by quantifying the attack status, attack intensity, and the mitigation cost-effectiveness.

However, most existing DDoS detection approaches struggle to meet these requirements.

Second, most learning-based approaches lack adaptability. Their performance is heavily dependent on the coverage and

Manuscript received 26 April 2022; revised 8 May 2023; accepted 29 June 2023. Date of publication 3 August 2023; date of current version 11 July 2024. This work was supported by the Science and Technology Directorate of the United States Department of Homeland Security under Grant D15PC00204. *(Corresponding author: Jun Li.)*

Yebo Feng and Jun Li are with the Department of Computer Science, University of Oregon, Eugene, OR 97403 USA (e-mail: yebof@cs.uoregon.edu; lijun@cs.uoregon.edu).

Devkishen Sisodia is with the Department of Computer Science and Software Engineering, California Polytechnic State University, San Luis Obispo, CA 93407 USA (e-mail: dsisodia@cs.uoregon.edu).

Peter Reiher is with the Department of Computer Science, University of California, Los Angeles, CA 90095 USA (e-mail: reiher@cs.ucla.edu).

Digital Object Identifier 10.1109/TDSC.2023.3301293

applicability of the training data. Considering that DDoS attacks are diverse and network traffic regarded as DDoS in one environment might be considered legitimate in another (and vice versa), few of the current learning-based approaches can effectively adapt a DDoS detection model trained in one environment to fit in a new network environment. This limitation leads to poor detection accuracy or the need for lengthy retraining.

To address these gaps, we design a learning-based DDoS detection and classification approach that is not only effective, but also explainable and adaptable. Specifically,

1) With network traffic flows summarized into traffic profiles, our approach can detect an DDoS attack (i.e., detection) and identify DDoS sources (i.e., classification) accurately and quickly. To detect DDoS, it enhances the k-nearest neighbors (KNN) algorithm to place traffic profiles into different regions into the searching space and can categorize traffic profiles to be benign or malicious and detect if the current traffic profile corresponds to a DDoS attack. Furthermore, to improve the efficiency of the detection process, it introduces a k-dimensional (KD) tree algorithm to convert the KNN detection model into a semi-decision tree, which significantly reduce the time complexity of traffic monitoring to $O(d)$ in most cases, where d is the depth of the semi-decision tree. If a DDoS attack is detected, to identify DDoS sources, it will sort the traffic sources (i.e., senders' IP addresses) based on risk levels to minimize collateral damage, and iteratively identify and remove the malicious IP addresses until the traffic profile returns to a benign position in the KNN searching space.
2) Our approach is highly explainable, characterized by its *transparency*, *traceability*, and *heuristic attributes*. These attributes enable the generation of intuitive explanatory information, allowing network administrators to easily understand and act upon them. Upon detecting a DDoS attack, our approach not only sends an alert message but also provides a *risk profile*, a *visual detection model*, and a *status graph* to elucidate the attack. The risk profile represents the shortest euclidean distance from the current traffic profile to a benign region in the KNN search space, assisting network administrators in quantifying the attack's magnitude and the associated mitigation costs. The visual detection model clarifies the detection logic, network context, and root causes by illustrating the relative distances from the current traffic profile to illegitimate and legitimate groups. Generated using principal component analysis (PCA) projection, the status graph concisely and intuitively depicts the attack stage, intensity, and cost-effectiveness of mitigation efforts.
3) Our approach is adaptable in that the detection and classification model derived in one environment can port to another environment without re-training. It allows direct modifications on the KNN searching space and enables users to leverage a variety of prior knowledge to evolve the detection model.

We evaluated our approach in both simulated environments and the real world. We first trained and evaluated our approach with representative DDoS datasets in simulation environments. The results indicate that our approach can achieve an accuracy of 0.956 and a recall of 0.920 even when detecting application-layer DDoS attacks. We then conducted a human subject study with questionnaire surveys to evaluate its explainability. The results demonstrate that the explanatory outputs can effectively help users understand not only the intensity, stage, and confidence level of the attack, but also can help them make suitable mitigation strategies quickly. Furthermore, as this approach is easily adaptable to new environments, we transferred the model (with merely some measurement data as input) to a real-world network environment at the Front Range GigaPoP (FRGP) [11], a regional IXP in USA. We successfully detected most of the real-world DDoS attacks from February 24 to May 21, 2020, which we verified with the IXP. The detection latency is also low—e.g., with a throughput of 100 Gbps, our approach can complete detection in around five seconds.

The remainder of this paper is structured as follows: Section II presents an overview of the related work, followed by a description of the threat and defense models in Section III. Subsequently, the method design is detailed in Section IV, and our approach is evaluated in Section V. Finally, we conclude the paper in Section VI.

II. Related Work

Using network traffic data to detect DDoS attacks is a technique that is widely used in the security community. From the perspective of operating principles, we can further classify the existing DDoS detection approaches into *statistical approaches*, *rule-based approaches*, *learning-based approaches*, and *soft-computing-based approaches*. We discuss the advantages and disadvantages of each approach in detail.

A. Statistical Approaches

Statistical approaches detect DDoS attacks by exploiting statistical properties of benign or malicious network traffic. These approaches are straightforward and dominated the early development of DDoS detection. Generally, these approaches build a statistical model of normal or malicious traffic and then apply a statistical inference test to determine if a new instance follows the model [12]. For example, D-WARD [13] uses a predefined statistical model for legitimate traffic to detect anomalies in the bidirectional traffic statistics for each destination with periodic deviation analysis. Chen [14] proposed a DDoS detection method based on the two-sample t-test, which indicates that the SYN arrival rate of legitimate traffic follows the normal distribution and identifies a DDoS attack by testing the distribution compliance. Zhang et al. [15] proposed a detection method by applying the Auto Regressive Integrated Moving Average model on the available service rate of a protected server.

Statistical approaches can provide interpretable results by outputting abundant metrics to describe the current network situation, such as shown in Fig. 1. These metrics primarily function as network measurements, assisting network administrators in grasping the network context. However, they are often not arranged in a concise and heuristic manner that would enable the identification of root causes and cost-effective mitigation

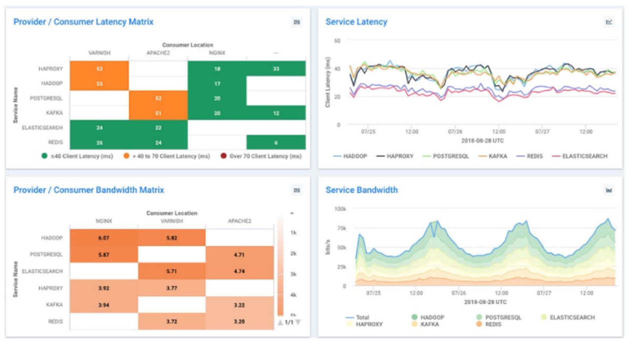

Provider / Consumer Latency Matrix
Service Latency
Provider / Consumer Bandwidth Matrix
Service Bandwidth

Fig. 1. Partial outputs of Kentik [16], a popular traffic monitoring tool that can detect and mitigate DDoS attacks.

strategies. As a result, skilled analysts are still indispensable for extracting valuable insights from these metrics to not only understand the importance of the alarms but also determine the appropriate course of action. Another limitation of statistical approaches is that as DDoS attacks evolve, traffic generated by sophisticated DDoS attacks may not always exhibit significant statistical deviations across various aspects. Consequently, traditional statistical DDoS detection methods might struggle to accurately identify modern DDoS attacks.

B. Rule-Based Approaches

Rule-based approaches formulate noticeable characteristics of known DDoS attacks and detect actual occurrences of such attacks based on those characteristics. For example, Net-Bouncer [17] detects illegitimate clients by conducting a set of legitimacy tests on the clients; If a client fails to pass these tests, it will be considered malicious until a particular legitimacy window expires. Wang et al. [18] detect DDoS with an augmented attack tree, which captures incidents triggered by DDoS traffic and the corresponding state transitions from the view of network traffic transmissions. Limwiwatkul et al. [19] detect ICMP, TCP and UDP flooding attacks by analyzing packet headers with well-defined rules and conditions. However, due to the growing diversity of DDoS attacks, rule-based approaches face challenges in summarizing and formulating the features of all possible attack types. Consequently, they are being gradually replaced by learning-based or soft-computing-based methods.

C. Learning-Based Approaches

Over the past few years, more and more researchers have begun to leverage machine learning to model, mitigate, and detect DDoS attacks (e.g., [5], [20], [21], [22], [23], [24], [25], [26], [27], [28]). Some of these methods (e.g., [29], [30], [31]) utilize unsupervised learning algorithms to distinguish anomalies from normal traffic, as such algorithms do not require training before detection. However, unsupervised-learning-based approaches are sensitive to the selected features and the background traffic. On the other hand, supervised-learning-based approaches may struggle to provide users with explainable results, as the prevalent machine learning algorithms (e.g., linear regression [32], multilayer perceptron [33], convolutional neural network [34], graph convolutional network [35], etc.) often resemble black boxes in their functionality. In real-world deployments, explainable results are critical for attack mitigation, because network administrators usually need to manually review the detection results to eliminate false positives and maintain the usability of their network infrastructure.

Recently, there has been a surge of efforts aimed at enhancing the explainability of machine learning algorithms. For example, Nguyen et al. [36] proposed a learning-based anomaly detection approach capable of informing users about the types of detected anomalies and the significant features contributing to the detection process. Ribeiro et al. [37] introduced Local Interpretable Model-agnostic Explanations (LIME), which offers insights into machine learning model predictions by generating locally interpretable explanations, enabling users to better comprehend the decision-making process of complex models. Additionally, Lundberg et al. [38] presented SHapley Additive exPlanations (SHAP), a unified method for explaining the output of various machine learning models. However, some of these methods have not been utilized for DDoS detection; certain explanations they offer may not be suitable for DDoS detection scenarios, or they may not fully satisfy the criteria for transparency, traceability, or heuristic characteristics.

In addition, the applicability of these machine learning algorithms highly depend on the training data and training environment, which means it is difficult to quickly transfer a detection model trained in one network environment to another network environment.

Therefore, although most learning-based approaches are usually accurate in detecting DDoS attacks, they are not easily deployable in real-world environments. As opposed to these previous learning-based approaches, our approach focuses on the explainability and adaptability.

D. Soft-Computing-Based Approaches

Soft computing is a term for describing the use of approximate calculations to provide imprecise but usable solutions to complicated computational problems. Such approaches match the general goal of DDoS detection, which is to identify attack sources while allowing only a few false positives and false negatives. Soft computing approaches can be an ensemble of statistical, rule-based, and learning approaches. For example, Jalili et al. [39] use statistical preprocessing to extract features from the traffic, and then utilize an unsupervised neural network to classify traffic patterns as either malicious or legitimate. Kumar et al. [40] utilize a resilient back propagation neural network as the base classifier, then propose RBPBoost to combine the outputs, and Neyman Pearson cost minimization strategy to generate the final classification decision. Shiaeles et al. [41] detect DDoS attacks based on a fuzzy estimator using mean packet inter-arrival times within 3-second detection windows. Just like learning-based approaches, soft-computing-based approaches also have the disadvantage of poor explainability, making them difficult to deploy in real-world scenarios.

III. Threat and Defense Models

In this section, we begin by presenting the threat models associated with DDoS attacks, followed by a description of the defense model of the proposed approach.

A. Threat Model

DDoS attacks are malicious efforts aimed at disrupting the normal operation of a targeted server, service, or network by inundating it with an overwhelming volume of internet traffic. These attacks are carried out by multiple systems, often compromised by malware and controlled by a single attacker known as a botmaster. The compromised systems, referred to as bots, constitute a network called a botnet. The primary objective of a DDoS attack is to render the target's resources inaccessible to legitimate users, resulting in downtime and potential financial or reputational harm.

Regarding attack methodologies, DDoS attacks can be categorized into three main types:

- *Volumetric attacks:* strive to overwhelm the target's bandwidth by generating an immense volume of traffic, impeding legitimate users from accessing the targeted service. Examples of volumetric attacks include reflector attacks, UDP floods, and ICMP floods [42].
- *Protocol attacks:* leverage vulnerabilities in network protocols to consume resources or cause network disruptions. For instance, SYN floods attack by targeting the TCP handshake; Ping of Death attacks function by transmitting oversized ICMP packets [43].
- *Application-layer attacks:* focus on certain applications or services, overloading them with seemingly legitimate requests. Such attacks demand fewer resources for execution but may pose greater challenges in defense. Examples include HTTP GET floods, Slowloris attacks, and DNS query floods [44].

DDoS attacks can cause significant harm to victims, leading to service disruptions, revenue loss, reputational damage, and increased security expenses. Therefore, it is crucial for organizations to implement strong security measures to minimize the impact of such attacks.

B. Defense Model

The defense model of the proposed approach operates as follows:

1) Initially, the network administrator deploys the proposed approach on the network to be secured. It is important to note that this network may differ from the one where the approach was trained.
2) The approach continuously monitors network traffic, identifying any DDoS attacks aimed at targets within the protected network.
3) Upon detecting a DDoS attack, the approach classifies the DDoS traffic and sends the classification results as mitigation rules to upstream routers or Internet service providers.
4) The mitigation rules typically include malicious IP addresses/IP prefixes or malicious traffic flows. These rules are then applied to network traffic to alleviate the DDoS attack, preventing it from reaching its intended victim.

1) Adaptability to the Network to Be Secured: The network requiring protection might not be the same as the one on which the approach was trained. This can occur when the approach is trained using a public dataset that may not accurately represent the specific network to be secured. Consequently, it is essential for the approach to rapidly adapt to the network in need of protection without necessitating extensive time, a large volume of training data, or numerous fine-tuning processes.

2) Minimizing Collateral Damage and Verifying Results: In the context of DDoS attacks, collateral damage refers to the unintended consequences of mitigation rules on legitimate traffic. If a rule inadvertently blocks a valid IP address, it can disrupt genuine traffic, potentially causing more harm than allowing malicious traffic to reach the intended target.

To minimize collateral damage, network administrators typically need to manually verify detection results before implementing them as mitigation rules (at steps 2 or 3). Several factors should be considered during the verification process, including:

- *Network Context:* Administrators should evaluate the network context, taking into account factors such as attack intensity, the number of attack sources, current network throughput, and more. This information is vital for understanding the attack's impact and the necessity of mitigation, allowing for appropriate planning and next steps.
- *Detection Logic:* Understanding the detection logic of the chosen approach is essential for administrators to determine the reliability of the results. Additionally, this information can help identify the root cause of the attack, aiding in the elimination of potential false positives. For example, during high-traffic periods, duplicate user requests may be misclassified as application-layer DDoS attacks (i.e., flash crowds). This type of misclassification can be avoided by quick verification.
- *Mitigation Cost-Effectiveness:* Since mitigation rules can lead to collateral damage and additional costs, administrators should weigh the cost-effectiveness of the proposed rules. In some cases, even when the network is under attack, the system may have enough redundancy to cope during periods of low activity. In such instances, administrators may opt not to apply mitigation rules to avoid unnecessary collateral damage.

Thus, the proposed approach should offer adequate explanations to aid administrators in verifying the results. Specifically, it needs to be *transparent* for assessing the network context, *traceable* for understanding the detection logic, and *heuristic* for evaluating the mitigation cost-effectiveness and formulating an appropriate plan.

IV. Design

In our approach, DDoS detection and classification occurs at the victim end, on a vantage point that sees all the traffic to and from the victim. It can stream explanations along with detection

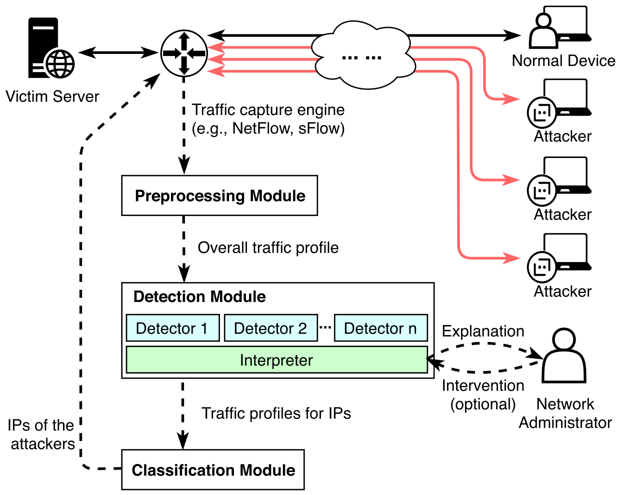


Fig. 2. Operational model of the proposed approach.

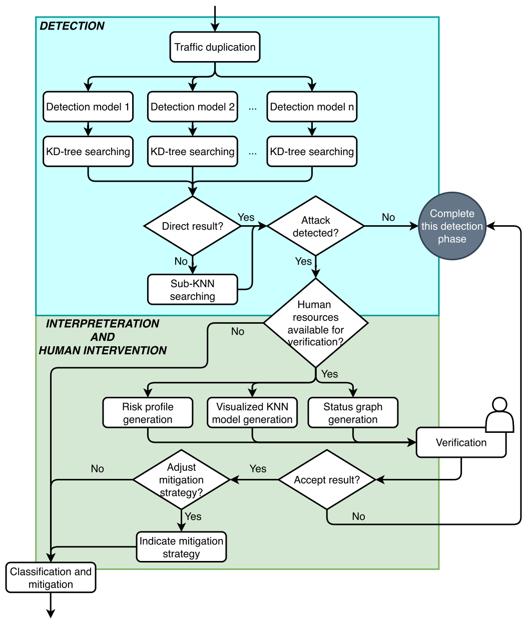


Fig. 3. Workflow of the detection phase.

results to the network administrator and allow interventions to the detection pipeline. Fig. 2 illustrates our approach's operational model. It has three components: the preprocessing module, detection module, and classification module.

First, the preprocessing module inputs packet or flow-level traffic data from the router that runs widely used traffic capture engines, such as NetFlow [45] and sFlow [46]. It monitors the traffic in batches. Each batch is a uniform time bin, t, which is also the most basic detection unit of our approach. In our implementation, we set each batch as 5 seconds. During each batch, the preprocessing module extracts features from the input data stream to form different types of overall traffic profiles. A traffic profile can be denoted as $\boldsymbol{s}$, with $\boldsymbol{s} = \{f_1, f_2, f_3, \ldots, f_n\}$, where f_n denotes the value of the n-th feature during a batch t. The features in $\boldsymbol{s}$ depend on the detectors we use, as each detector may need a different traffic profile with different features.

Our approach then works in two phases: the detection phase (illustrated in Fig. 3) and classification phase. In the detection phase, the detection module detects whether the network is under a DDoS attack. To provide comprehensive protection to the victim, our approach can employ multiple detectors, with each focusing on certain types of DDoS attacks. Once a DDoS attack is detected, the detection module outputs both detection results and explanations to ongoing attacks. The network administrators can review and verify the detected attack according to the explanatory information, thereby choosing to intervene in the attack defense procedure or allow our approach to automatically deal with the attack. In the end, the classification phase begins by pinpointing the IP addresses of attackers for future actions. In this phase, the classification module generates a traffic profile $\boldsymbol{p}$ for every individual IP address and classifies traffic at a fine granularity according to IP traffic profiles.

A. Detection Phase

The goal of the detection phase is to determine whether a DDoS attack is present according to the current traffic profile $\boldsymbol{s}$. We use the KNN algorithm [47] to achieve the goal, as this algorithm is straightforward and reliable. The KNN algorithm is a non-parametric method used for classification, which finds the k nearest neighbors of the traffic profile $\boldsymbol{s}$ and uses their classifications to vote for the label of $\boldsymbol{s}$. Users can also choose to build multiple KNN detection models to detect a variety of DDoS attacks, as Figs. 2 and 3 show.

In our implementation, we built four distinct detection models to identify TCP SYN floods, ICMP floods, UDP reflection and amplification attacks, and application-layer attacks, respectively. Each model utilizes different features and training data. The rationale behind constructing multiple KNN models to address different attacks, rather than developing a single complex KNN model, is to circumvent the curse of dimensionality [48] and overfitting. A detection model capable of handling various types of attacks generally needs to process data in high-dimensional spaces since it must encompass all the essential features of each individual attack. Nonetheless, an increase in the dataset's dimensions can render the search space sparser. Consequently, we would require significantly more training data to cover the search space; otherwise, the detection model's accuracy would be unsatisfactory. To overcome this issue, we build multiple KNN models to cover different attacks, with each model using only a few features.

Users are able to adjust the voting mechanism of the KNN algorithm to get detection results with higher confidence, thereby reducing the number of false alarms in real deployments. More specifically, our approach labels the current traffic profile as malicious if more than ρ of the k nearest neighbors in the KNN searching space are malicious. We usually assign a value to ρ that is greater than 0.5. A larger value of ρ corresponds to a less stringent detection standard. For instance, in our evaluation, we set ρ to 0.5 to diminish the false positive rate.

However, the KNN algorithm has a notable drawback. Although the model training time is minimal, the prediction requires a time complexity of $O(nlogn)$ to complete, as it needs

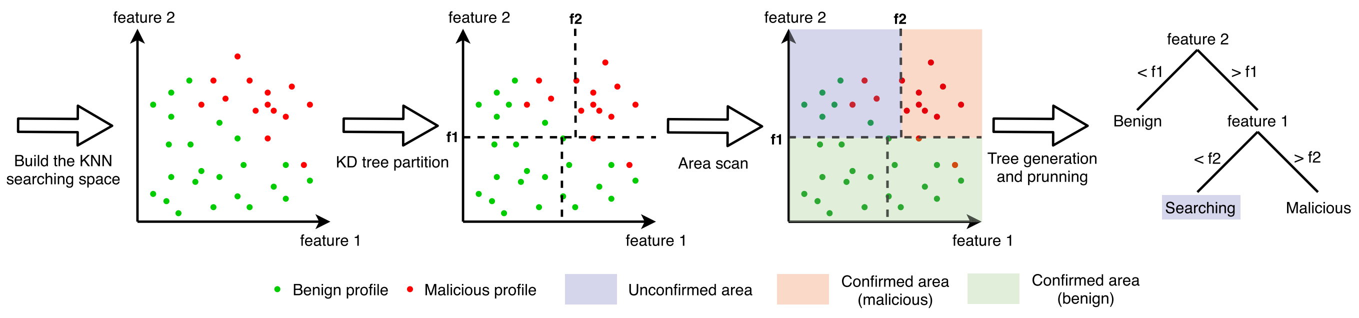


Fig. 4. DDoS detector with the modified KNN algorithm and KD tree.

to enumerate the data points in the search space to find the k nearest neighbors. To address this issue, we leverage the KD tree [49] to partition the search space, thus reducing the number of data points to enumerate. With the KD tree, when an incoming traffic profile arrives, we only need to search a sub-area to predict the result. Fig. 4 illustrates a simple example where only two features are included in the training and prediction process.

Furthermore, according to our experimental results, most DDoS profiles exhibit relatively large differences compared to legitimate traffic profiles. This leads to an intriguing observation that most of the search areas partitioned by the KD tree contain either benign traffic profiles or malicious traffic profiles. As shown by the red and green areas in Fig. 4, we define a search area as a confirmed area if one type of traffic profile dominates the area and the number of any other type of traffic profile is smaller than ρk. If the current traffic profile $\boldsymbol{s}$ falls within a confirmed area, we can directly label the profile $\boldsymbol{s}$ with the identity of the confirmed area without conducting any KNN queries. As a result, we transform the original KNN query process into a semi-decision tree. The detection module will only trigger the search for nearest neighbors when the traffic profile $\boldsymbol{s}$ falls within an unconfirmed area. If anomalies do not occur frequently, this semi-decision tree data structure can reduce the time complexity for traffic monitoring to nearly $O(d)$, where d is the depth of the tree.

However, the use of the KD tree may lead to a slight decrease in detection accuracy. This is because the search space is partitioned into multiple sub-areas, which may result in inaccurate results when the traffic profile $\boldsymbol{s}$ falls on the boundary of two sub-areas. Nonetheless, for most DDoS attacks, legitimate traffic profiles have relatively large distances from malicious traffic profiles, leading to a significant margin between the two types of traffic profiles, thereby minimizing the impact caused by the KD tree. Moreover, by employing multiple models to detect different types of DDoS attacks, we ensure clear decision-making for each detection model, which further minimizes the impact brought by the KD tree on detection accuracy.

B. Explainability & Manual Intervention

Once our approach detects a DDoS attack, it not only outputs an alert message, but also employs an interpreter (as shown in Fig. 2) to export transparent, traceable, and heuristic explanatory information to explain and quantify the attack. Such information includes a *risk profile*, a *visualized KNN model*, and a *status graph*. According to these outputs, network administrators can know the attack type, detection logic, intensity, status, confidence level of the alarm, and the cost of mitigations. Unlike some statistical approaches that provide too many metrics that can easily overwhelm network administrators, our method aims to output concise information and intuitive explanations with the help of appropriate visualizations. With a small amount of training, network administrators can understand the current situation within seconds on the basis of the interpreter's outputs, and therefore are able to quickly make manual interventions to the detection decision. Furthermore, network administrators can choose to either reject or approve the detection results. Of a particular note is that this manual intervention is optional. If the administrator does not intervene within a certain amount of time, the system will automatically execute the decisions of detectors.

1) Risk Profile: The risk profile Δ (where $\Delta = (m, \delta)$) is a tuple that provides the network administrators with a quantified and traceable summary about the current attack, indicating the primary cause and intensity, which meets the traceability requirement in the paradigm of explainability. Here, m is the name of the feature in the traffic profile $\boldsymbol{s}$ that primarily causes the DDoS attack. This attribute helps the network administrator determine the attack type. For example, if m is the "number of inbound ICMP packets", the victim is likely facing an ICMP attack and being overwhelmed by abundant incoming ICMP packets. δ is the smallest value by which feature f_m needs to be reduced to make the traffic profile $\boldsymbol{s}$ move to a benign position. In other words, δ is the shortest distance on f_m from the current traffic profile to a legitimate traffic profile in the KNN searching space. For example, $\Delta = (''number\ of\ inbound\ ICMP\ packets'', 8500)$ means the victim is currently under an ICMP flooding attack and we need to eliminate at least 8500 inbound ICMP packets per five seconds to mitigate the attack.

To figure out Δ for a given traffic profile $\boldsymbol{s}$ that has been labeled as DDoS attack by a detector D, we need to first find the closest benign traffic profile $\boldsymbol{l}$ in the KNN searching space of D. To achieve this, we conduct a breadth-first search. Then, we normalize $\boldsymbol{s}$ and $\boldsymbol{l}$ to ensure that features belonging to both profiles are directly comparable. In the end, we use 1 to calculate

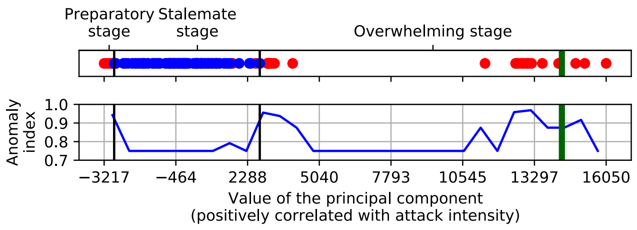


Fig. 5. Status graph of a SYN flood detector. Red dots represent attack traffic profiles. Blue dots represent legitimate traffic profiles. The dark green vertical line represents the current traffic profile.

δ and m.

$$\begin{aligned} \boldsymbol{s_\Delta} &= \boldsymbol{s}_{normalized} - \boldsymbol{l}_{normalized}, \\ m &= max(\boldsymbol{s_\Delta}).FeatureName, \\ \delta &= max(\boldsymbol{s_\Delta}). \end{aligned} \tag{1}$$

In a few cases, the interpreter may find multiple risk profiles from multiple detectors, which means $\Delta = \{(m_1, \delta_1), (m_2, \delta_2), \ldots, (m_n, \delta_n)\}$. We consider that the victim is facing either flash crowds or severe hybrid attacks under this circumstance, as the traffic volume significantly exceeds the infrastructure's capacity in multiple aspects. Here, flash crowds are large surges of legitimate traffic focusing on specific sites on the Internet over a relatively short period of time [50].

2) Visualized KNN Model: To fulfill the transparency requirement and provide network administrators with a clear understanding of the detection model, network context, and detection logic, the interpreter will visualize the KNN detection model in addition to the detection results. As the training and input datasets are usually of high dimensionality, the interpreter will only include three most important features of the datasets to draw a three-dimensional plot. Besides, the network administrator can choose to change the visualized features to inspect the situation from different aspects.

Such a visualized KNN model is informative. From the visualization, network administrators can observe relative distances from the current traffic profile to benign and malicious groups. According to this information, network administrators can obtain intuitive understandings regarding the detection logic, attack severity, and victim status. We further evaluate the explainability of the visualized KNN model in Section V-C.

3) Status Graph: To facilitate rapid decision-making by network administrators based on current conditions and the cost-effectiveness of mitigation measures, the interpreter generates a status graph that provides a concise and intuitive representation of the attack stage, intensity, and confidence level of the alarm.

Fig. 5 shows a status graph example. It consists of two subplots. The upper one uses principal component analysis (PCA) [51] to map the training and input datasets to a one-dimensional space. PCA is a technique widely used for dimensionality reduction by projecting each data point onto only the first few principal components to obtain lower-dimensional data, while preserving as much of the data's variation as possible. More specifically, for a k-dimensional DDoS training dataset $\boldsymbol{D} \in \mathbb{R}^{N\times k}$, the interpreter uses PCA to learn a linear transformation shown in 2.

$$\boldsymbol{T}_1 = \boldsymbol{D}\boldsymbol{W}_1,\ \boldsymbol{T}_2 \in \mathbb{R}^{N\times 1}. \tag{2}$$

Then, for the incoming traffic profile $\boldsymbol{s}$, we use 3 to map it onto a two-dimensional space.

$$\boldsymbol{r} = \boldsymbol{W_1^T}\boldsymbol{s},\ \boldsymbol{s} \in \mathbb{R}^{1\times k},\ \boldsymbol{r} \in \mathbb{R}^{1\times 1}. \tag{3}$$

In the end, our approach visualizes this one-dimensional dataset $\{\boldsymbol{r}\} \cup \boldsymbol{T}_1$, labeling DDoS traffic profiles, legitimate traffic profiles, and the input traffic profile with different colors. In other words, it is a reduced-dimensional KNN model. Network administrators can quickly learn relative spatial relationships between the current traffic profile and attack/legitimate traffic profiles from this plot.

The subplot below illustrates the anomaly index κ. This value indicates the confidence level of the detection result. The closer this value is to one, the more likely it is that the detected attack is a true positive. Since all of the alarms are detected by the KNN model, the base value of κ is equal to ρ. Then, our approach utilizes a window to move from left to right in the upper subplot, checking the number of attack and legitimate traffic profiles within the window to calculate κ.

$$\kappa = \rho + (1-\rho)\frac{n_m}{n_i + n_m}. \tag{4}$$

4 shows the calculation of the anomaly index κ, where n_m denotes the number of malicious traffic profiles within the window and n_i denotes the number of legitimate traffic profiles within the window.

In addition, by analyzing the training data, our approach divides the status graph into three stages from left to right:

- *Preparatory stage:* the attack is still in its infancy. Its intensity is low. The network administrator can choose to ignore this attack if conducting conservative defensive measures.
- *Stalemate stage:* the attack is still under the infrastructure's capacity, but it is starting to cause a noticeable impact on the network. Network administrators should mitigate the attack if conducting rigorous defensive measures. However, network administrators can still ignore this attack if they are more concerned about collateral damage caused by mitigation.
- *Overwhelming stage:* the attack is overwhelming the network, the network administrator should immediately take mitigation measures to protect the accessibility of the network.

Network administrators can know the attack status by observing which area the current traffic profile falls in.

From the example in Fig. 5, we can discern from the status graph that the detected attack is in the overwhelming stage. The current traffic profile is much closer to malicious groups. Moreover, both the attack intensity and anomaly index are high. Therefore, network administrators should immediately take measures to mitigate this attack.

C. Phase Two: Classification

The objective of the classification phase is to differentiate malicious IPs from benign ones, and output the malicious IPs for DDoS traffic filtering. It is important to note that the classification module will only be activated after some anomalies have been detected in the detection phase.

The design philosophy of the traffic classification is that the traffic profile $\boldsymbol{s}$ is currently in a malicious position, and we need to restrict the traffic from the most suspicious IP addresses so that the traffic profile can move to a safer position in the KNN searching space.

We begin the classification phase by building a traffic profile $\boldsymbol{p}$ for each IP that appeared during the attack. The profile $\boldsymbol{p}$ should have the same attributes as the overall traffic profile $\boldsymbol{s}$. The only difference is that the values of features in $\boldsymbol{p}$ are calculated from the traffic of each individual IP, while the values of features in $\boldsymbol{s}$ are calculated from the overall traffic in the network. Afterwards, we sort the IPs in decreasing order of the risk degree (i.e., a number that indicates how suspicious an IP is). According to the risk profile Δ ($\Delta = (m, \delta)$) we obtained from the DDoS detection phase, we define the risk degree of an IP as $f_m^{(\boldsymbol{p})}$. Finally, we conduct traffic filtering on IP addresses such that the overall traffic profile can move to a benign area.

However, legitimate IPs may sometimes have large risk degrees as well. Classifying the IP addresses only according to the risk degree may cause significant collateral damage. To address this issue, we also need to minimize the impact on other features of the overall traffic profile $\boldsymbol{s}$ when determining the malicious traffic sources. We consider this as an optimization problem with two constraints, which can be demonstrated by 5. Here, G denotes the complete set of IP addresses we have seen in the network during the DDoS attack, G_m denotes the set of malicious IP addresses that the classification program will output for future actions, and $p^{(i)}$ denotes the traffic profile of the ith-IP.

$$\underset{G_m}{\text{argmax}}\, f(G, G_m) = \sum_{g\in G, g\notin G_m} \sum_{i\in g} \left\|\boldsymbol{p}^{(i)}\right\|_2 = \sum_{g\in G, g\notin G_m} \sum_{i\in g} \sqrt{\sum_{k=1}^{n} \left|f_k^{\boldsymbol{p}^{(i)}}\right|^2}, \quad (5)$$

$$\text{subject to: } \sum_{g\in G_m} \sum_{i\in g} p_m^i \geq \delta,$$

$$G_m \subseteq G. \quad (6)$$

6 shows two constraints: (1) after eliminating all the traffic from malicious IP addresses (set G_m), the overall traffic profile should be reduced by at least δ on f_m in the KNN searching space; (2) the malicious IP set G_m should be a subset of the complete IP set G.

Deriving the optimal solution of this optimization problem is expensive, especially when the network we are monitoring is at the ISP-level. Hence, we designed Algorithm 1 to obtain a near-optimal solution G_m efficiently. Since the time complexity of sorting the IPs according to the risk degree is $O(nlogn)$, the algorithm conducts the grid partitioning on the searching space to accelerate the IP classification. Then, we need to eliminate IP addresses along the m axis and minimize impacts on other features at the same time. With this grid configuration, we can always find a corner grid g_m that has the largest value on feature m but also has the smallest values on irrelevant features. The classifier considers the grid g_m as the most suspicious grid and gives it the highest priority in classification. Afterwards, the algorithm sorts the remaining grids in decreasing order of feature m and increasing order of other features. Finally, the algorithm eliminates IPs grid-by-grid in such order until the overall traffic profile returns to the benign area. Fig. 6 illustrates an example of such procedure.

Algorithm 1: Recognition of Malicious IPs With Grid Sorting.

```
 1: input: risk profile $\Delta = (m, \delta)$
 2: input: complete IP set $G$
 3: initialize set $G_m$ to store the malicious IP addresses
 4: grid partitioning: $G = \{g_1, g_2, g_3, \ldots, g_n\}$
 5: $G.sort()$    ▷ in decreasing order of feature $m$ and
    increasing order of other features
 6: for $g$ in $G$ do
 7:    $G_m.add(g.items())$
 8:    $val \longleftarrow \sum_{i\in g} f_m^{p^{(i)}}$
 9:    $total\_eliminated \longleftarrow val + total\_eliminated$
10:    if $total\_eliminated >= \delta$ then
11:       return $G_m$
12:    end if
13: end for
```

D. Adaptability

The proposed approach offers superior adaptability compared to other learning-based methods. When deploying a pre-trained detection model in a new network environment, users are not required to retrain the model for a suitable fit. Instead, they can leverage a variety of prior knowledge to evolve the model, thereby enhancing its robustness across different environments.

Here, we assume the user will have some type of limited information about the new network environment as prior knowledge. Such information includes the network traffic measurements or link bandwidth information about the network environment, some training samples for online learning, and incomplete threshold values for DDoS detection. Any type of the above information can evolve the detection model and help the model adapt to the new environment.

1) Mapping Via Traffic Measurement: Assuming that we have the network traffic measurement results about the new network environment, we can normalize the KNN searching space from the trained environment to the new environment according to the traffic distributions of the two networks. The easiest way to do this is by using min-max normalization for the conversion process.

$$l = max(\boldsymbol{D}_{new}[:, i]) - min(\boldsymbol{D}_{new}[:, i])$$

$$\widehat{\boldsymbol{D}}[:, \boldsymbol{i}] = l \cdot \frac{\boldsymbol{D}[:, i] - min(\boldsymbol{D}[:, i])}{max(\boldsymbol{D}[:, i]) - min(\boldsymbol{D}[:, i])} \quad (7)$$

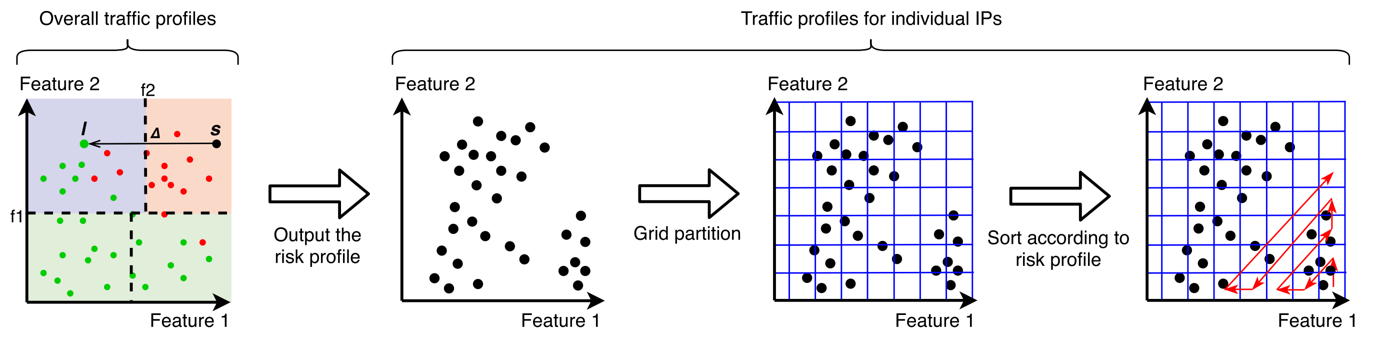


Fig. 6. Example of the classification process, where the classification module reads the risk profile, partitions the searching space, and find the malicious IP set G_m grid-by-grid.

Algorithm 2: Integration With Existing Rules.

```
1: **input:** existing rule table $T$ as a stack
2: **input:** detection model $D$    ▷ $D$ is a semi-decision tree
3: **while** $T$ is not empty **do**
4:   $r \longleftarrow T.pop()$
5:   **if** $D(r.condition)$ exists and overlaps with searching
     area set $S$ **then**
6:     remove overlapped areas from $S$
7:     $T.push(r)$
8:   **else if** $D(r.condition)$ exists **then**
9:     $D.update(r)$
10:  **else**    ▷ $D(r.condition)$ not exists
11:    $D.root.rightChild \longleftarrow D$    ▷ right child will be
       called when not satisfying the condition
12:    $D.root \longleftarrow r.condition$
13:    $D.root.leftChild \longleftarrow FILTER$ action
14:  **end if**
15: **end while**
16: **return** $D$
```

7 shows the conversion process, where $\boldsymbol{D}$ denotes the original training dataset and $\boldsymbol{D}_{new}$ denotes the sampled traffic from the new network environment. By mapping the original training data to the new network environment, our approach is able to conduct DDoS detection without retraining or re-collecting any new training data.

2) Online Updating for KD-Tree: If the traffic monitoring system can obtain labeled traffic with the system running, we can conduct online learning on the proposed detection model, thus making it gradually fit a new environment. The KNN algorithm does not require training, making it very suitable and efficient to conduct online learning. However, the KD-tree, along with the confirmed areas, needs to refresh to reflect new knowledge. We can control the program to update the classifier only during the idle times to reduce the performance impact on the detection system. Nevertheless, the time complexity of refreshing the model is only O(n).

3) Integration With Existing Thresholds/Rules: In certain situations, network administrators may already have imperfect detection rules (e.g., threshold-based rules) tailored to their network environment. Below are a few examples of such rules:

```
if (traffic.packets_per_second >
2_000_000
  or traffic.kbs_per_second > 1_800_000
  or traffic.in_out_ratio > 80
  or traffic.external_ips > 15_000):
  alert()
else: pass
```

Network administrators can integrate our approach with existing rules to enhance DDoS protection efficacy without disrupting the current detection logic or significantly increasing the rule budget. Since the pre-trained DDoS detection model is a semi-decision tree, users can incorporate existing detection rules into the pre-trained model by modifying the tree structure. Algorithm 2 illustrates an example procedure for the integration, where the existing detection rules have higher priority. Users can also specify different decision priorities based on the current situation.

V. Evaluation

In this section, we assess our approach from various perspectives. We tested our approach not only in simulated environments using multiple publicly-available DDoS datasets (Section V-B), but also deployed it at FRGP [11], a regional IXP in Colorado State, to examine its adaptability and usability in real-world scenarios (Section V-D). Additionally, we conducted a questionnaire survey to quantitatively evaluate the explainability of our approach (Section V-C).

A. Features & Training Data

Our learning-based approach requires labeled training data as input in order to build the detectors for each attack type. Therefore, we picked several representative DDoS datasets from public repositories and captured traffic in real-world environments to train and test our approach. Table I shows the public datasets we used and the types of attacks they contain. These datasets and our captured traffic cover volumetric attacks (e.g., ICMP flood, UDP reflection and amplification attacks), protocol attacks (e.g., TCP SYN flood attacks), and application-layer attacks (e.g., HTTP flood, Slowloris, etc.). We separately trained four DDoS detection models using the datasets, with one dedicated to TCP SYN flood, another to ICMP flood, a third one for UDP reflection and amplification attacks, and a final one for

TABLE I
DATASETS FOR TRAINING AND TESTING

Dataset Name	Format	Size	Attack Type	Background Traffic	Used For
DARPA 2009 DDoS [52]	pcap	1.09 GB	TCP SYN flood attack	✓	Training & Testing
CAIDA 2007 DDoS [53]	pcap	12.08 GB	ICMP flood attack		Training & Testing
FRGP NTP Flow Data [54]	Argus flows	1.60 TB	NTP reflection attack	✓	Training & Testing
DDoS Chargen 2016 [55]	flow-tools	74.05 GB	UDP reflection and amplification attacks	✓	Training & Testing
FRGP Colorado Traffic [11]	FlowRide & NetFlow	$>$ 5.00 TB	Various	✓	Testing

TABLE II
FEATURES WE UTILIZE FOR DETECTING AND CLASSIFYING DIFFERENT CATEGORIES OF DDOS ATTACKS

Attack Type	Features We Use
TCP SYN flood — *protocol attack*	# of inbound TCP packets / # of outbound TCP packets, # of TCP packets, # of inbound SYN packets, # of outbound ACK packets, # of inbound ACK packets
ICMP flood — *volumetric attack*	# of inbound ICMP packets / # of outbound ICMP packets, # of ICMP packets, # of inbound echo requests, # of outbound replies (destination unreachable)
UDP reflection & amplification attack — *volumetric attack*	# of inbound UDP bytes / # of outbound UDP bytes, # of UDP bytes, # of inbound UDP packets / # of outbound UDP packets, # of UDP packets
HTTP GET flood, Slowloris, DNS query attack, etc. — *application-layer attack*	# of inbound bytes / # of outbound bytes, # of bytes, # of sessions, # of inbound packets / # of outbound packets, # of packets, avg packet interval

application-layer attacks. Together, these models can provide comprehensive protection to the victim server.

The training datasets come in various formats, ranging from packet-level pcap data to flow-level connection data. Since our approach operates at the flow level, we preprocess the data by converting the original datasets into traffic profiles tailored to different detection models with a granularity of five seconds. We also sampled a small portion (approximately 10%) of the data from the DDoS datasets as our testing datasets. These testing datasets were not used during model training phase. Moreover, we sampled network traffic from a router at FRGP to simulate legitimate background traffic, thereby complementing the dataset. The overall ratio of DDoS training data to legitimate background training data is 1:2.

As our approach works best with low-dimensional datasets, we selected the best feature sets based on univariate statistical tests. More specifically, we performed χ^2 tests to the data samples to retrieve only 4-6 best features. Table II enumerates the four sets of features we selected to train the four DDoS detectors. The most frequently used feature was the ratio of the inbound traffic volume to the outbound traffic volume. We found that the features listed in Table II are useful in identifying the majority of DDoS attacks.

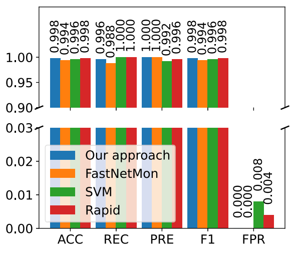
B. Detection & Classification Efficacy

To evaluate the detection and classification efficacy of our approach, we first built a simulation environment where a virtual switch continuously streams collected traffic to the proposed system. Such a simulation environment enables us to conduct convenient and efficient tests. During the evaluation, we simultaneously replayed legitimate traffic and a portion of the DDoS test traffic. We also dynamically adjusted the traffic volume during the test to effectively mimic real-world DDoS scenarios.

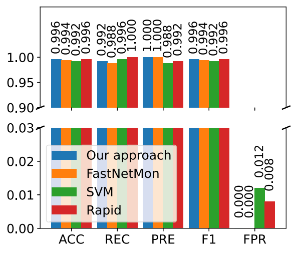
1) Detection Efficacy: For comparison tests, we utilized three additional DDoS detection approaches. One is a DDoS detection model based on a support vector machine (SVM) [56]. We trained this model using the same training data and features as shown in Table II. Another is FastNetMon [57], an open-source commercial DDoS detection program. This threshold-based DDoS detection approach is widely employed in small to mid-sized enterprises due to its high efficiency and accuracy. Lastly, we included Rapid [27], a hybrid DDoS detection method that combines LSTM and multi-layer perceptron. The test dataset consists of at least 250 episodes of legitimate traffic traces and at least 250 episodes of traffic traces with attacks. An episode is the most basic detection unit, containing more than five seconds of replayed network traffic.

Fig. 7 illustrates the comparison results for DDoS detection under simulated environments. For both SYN flood and ICMP flood attacks, all the three approaches can achieve decent detection efficacy. As for UDP and application-layer attacks, although Rapid and the SVM-based approach are slightly superior to our approach in terms of recall scores, they perform worse in terms of the false positive rates. A low false positive rate is essential for the detection system's usability, as a high number of false alarms will either cause too much collateral damage or force network administrators to ignore the detection results. Thus, when accuracies are similar, users tend to choose the approach with a significantly lower false positive rate. Compared with FastNetMon, our approach has a similar false positive rate. However, our approach is superior to FastNetMon in terms of accuracy and recall score.

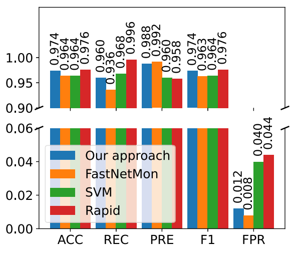
We also halved the training data, resulting in a 1:4 ratio between the DDoS training data and legitimate background training data, to assess the detection efficacy in the presence of unbalanced and insufficient training data. Fig. 7(e) and (f) present the results. We can see that when the training data is unbalanced, FastNetMon works significantly better than other approaches, as it is a threshold-based approach. Among the



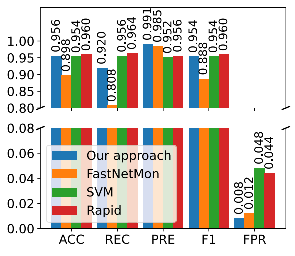
(a) SYN flood.



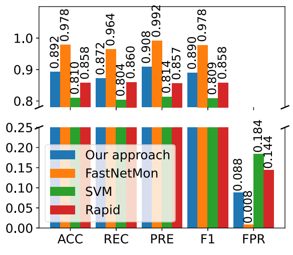
(b) ICMP flood.



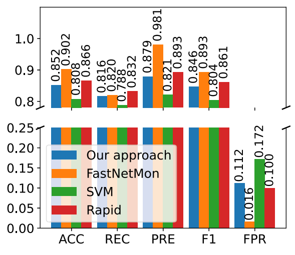
(c) UDP flood.



(d) Application-layer DDoS.



(e) Volumetric and protocol DDoS (unbalanced training data).



(f) Application-layer DDoS (unbalanced training data).

Fig. 7. Comparison results of DDoS detection efficacy (ACC: Accuracy, REC: Recall, PRE: Precision, F1: F1 score, and FPR: False positive rate).

other approaches, our method demonstrates superior detection efficacy compared to the SVM-based approach and exhibits comparable efficacy to Rapid.

2) Classification Efficacy: As for the traffic classification, we first replayed a collected network traffic dataset in a Mininet-based [58] network environment. This dataset consists of 25 minutes of network traffic with both DDoS attack and legitimate packets. Then, we ran FastNetMon and the proposed approach respectively, conducting mitigation on malicious IP addresses reported by them throughout each process. Simultaneously, we observed the network situation to evaluate the classification efficacy. To ensure a fair procedure, we did not intervene in the detection process during evaluation.

Fig. 8 shows the classification efficacy results, where the y-axis indicates the number of packets. By mitigating all the traffic from the attackers classified by the two approaches, we can see our approach can eliminate more malicious traffic than FastNetMon. The only drawback of our approach is that the classification will only be triggered when an attack is detected. After proceeding with mitigation, once the traffic profile is no longer labeled as malicious, our approach will stop classifying

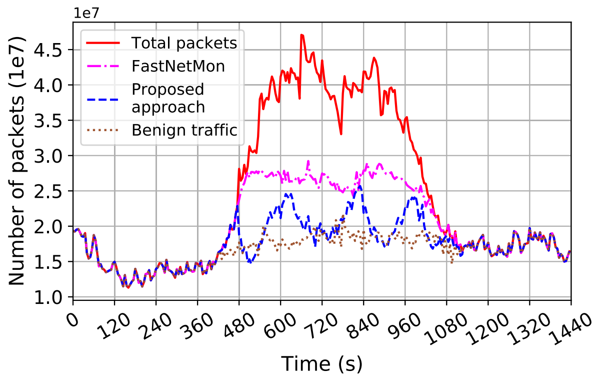


Fig. 8. Efficacy of DDoS traffic classification.

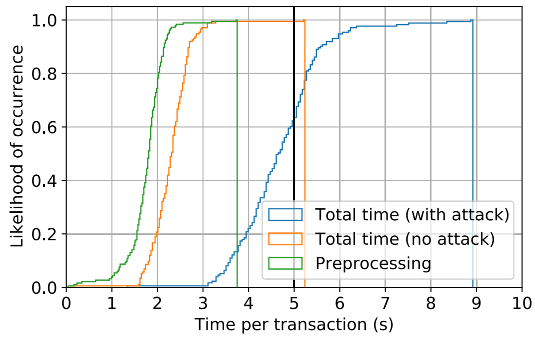


Fig. 9. Cumulative step histograms of processing time (tested with 100 Gbps flow-level traffic).

IP addresses as malicious, and only begin classification again as soon as the traffic profile is labeled malicious again. This explains the periodic fluctuations on the number of packets for our approach as seen in the figure.

3) Timeliness: We measured the runtime of our approach on a 100 Gbps link (please refer to Section V-D for details of the link) and presented the results in Fig. 9. This figure shows three cumulative step histograms, illustrating the runtime for pre-processing a batch of traffic (five seconds), monitoring a batch of legitimate traffic, and monitoring a batch of traffic with attacks, respectively. Here, the runtime for monitoring legitimate traffic consists of the time consumption of pre-processing and detection; the runtime for monitoring traffic with attacks consists of the time consumption of pre-processing, detection, and classification.

From the figure, we can see that the runtime is short when there are no attacks present, considering that the program has a five-second time window to operate. Moreover, as the detection model is a semi-decision tree, it will directly output the results without conducting any KNN queries if the traffic profile is situated in a confirmed benign area. Thus, the majority of time is spent on traffic pre-processing when monitoring only legitimate traffic. When there is a considerable amount of incoming DDoS traffic, the runtime almost doubles since fine-grained IP classification is time-consuming. Fortunately, when an attack is detected, the system does not need to complete the calculation within five seconds to catch the next batch. The top priority at the

TABLE III
BASIC INFORMATION OF THE QUESTIONNAIRE SURVEY

Composition of the participants	
Participants with DDoS-related expertise	15
Participants with security backgrounds but not DDoS-related expertise	4
Participants without security backgrounds	4
The total number of participants	23
Basic information of the questionnaire survey	
Number of questions	25
Approximate time to explain the usage of our approach (min)	15
Approximate time to complete the survey (min)	30

time an attack is detected is to mitigate the attack, and therefore, an increased delay in classification is still acceptable.

In conclusion, our approach is efficient in detecting and classifying DDoS traffic. With delays of around two seconds during idle time and five seconds during the DDoS peak, our approach is able to produce timely defense for victims.

C. Explainability

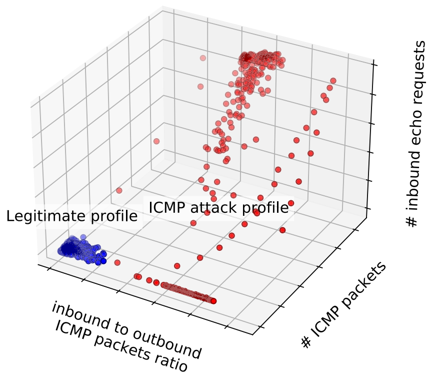
To evaluate the explainability of our approach, we conducted a questionnaire survey, which is a formal and effective method in Human-Computer Interaction (HCI) research [59], to collect feedback from participants and assess their understanding of the system's functionality and decision-making process.

We disseminated survey questionnaires to a range of security labs and individuals without a security background in the USA and China. In total, 23 people participated in the survey. Table III provides an overview of the participants' basic information as well as essential details about the questionnaire survey.

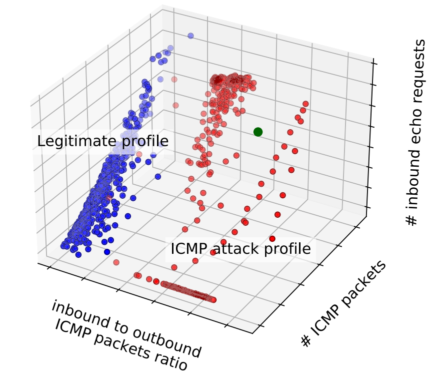
Before the questionnaire, we provided a brief introduction to the background knowledge, our design, and the output explanatory information. We then presented several examples of the outputs to demonstrate how they explain detected attacks and how to interpret them. Figs. 10 and 11 display a few examples from the questionnaire.

We proceeded to ask participants questions about the explainability of our approach, such as the ease of understanding the outputs, the intuitiveness of the visualizations, and whether the explanatory information met the design objective. Finally, we administered tests to assess participants' comprehension of the explanatory information for various attack types and their ability to make correct interventions. Specifically, we presented scenarios of different attack types detected by our approach and asked participants to interpret the explanatory information and recommend intervention measures for the next step under varying circumstances. Responses were collected anonymously to protect privacy and minimize bias.

Fig. 12 presents some key findings from our questionnaire evaluation. Although a few individuals questioned the explainability aspects of our approach, the majority of participants agreed that the risk profile helps users understand the root cause and quantify the attack. Additionally, the visualized KNN model provides an intuitive explanation of the network context, detection models, and detection logic for network administrators. The status graph illustrates the current attack stage, intensity, confidence level of the alarm, and mitigation cost-effectiveness,



(a) Before normalization. We can clearly see that the training data does not fit the network environment.



(b) Normalized according to the traffic distribution. The dark green dot is the current traffic profile for inference, whose risk profile $\Delta = (\text{"}number\ of\ inbound\ ICMP\ packets\text{"}, 52041)$.

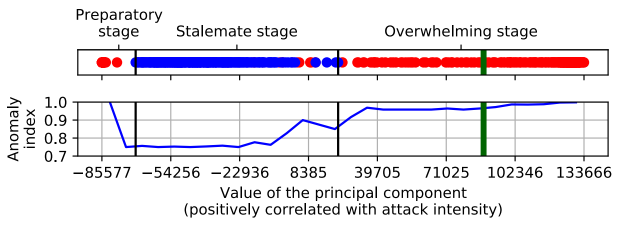
Fig. 10. Examples of visualized KNN detection models for identifying ICMP attacks.

ultimately guiding network administrators in making appropriate interventions. Hence, with respect to transparency, traceability, heuristic qualities, and learnability, our proposed method successfully accomplishes its intended design objectives.

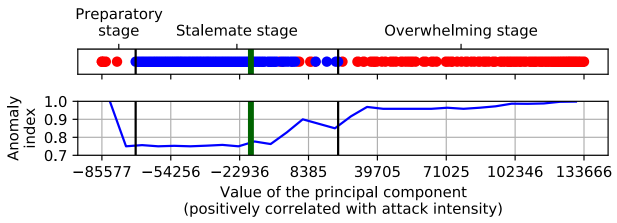
D. Case Study: A Real-World Deployment

In addition to the evaluation under emulation environments, we deployed our approach at several links in FRGP to further test its deployability and adaptability. This real-world deployment also provides a good opportunity to demonstrate how explanatory information can help network administrators adopt conservative tactics for eliminating false alarms.

1) Measures for Ethical Considerations: As the network traffic from FRGP contains private information of users and trade secrets of operators, we take effective measures to address possible ethical considerations. Data is collected by FRGP operators and their collaborators from a local educational institution on an ongoing basis. We formulate a Memorandum of Agreement (MoA) with FRGP operators and their collaborators to stipulate the correct usage and accessibility of the data. To protect the privacy of users and prevent data leakage, we set rigorous regulations for data analysis and storage. We list the regulations below:



(a) Status graph of an attack shown in Figure 10b. Network administrators should proceed with mitigation as this attack has a high intensity and is already in the overwhelming stage.



(b) A detected ICMP flooding attack. This attack is in the stalemate stage. Network administrators can either ignore this attack if following a conservative protection policy or proceed with mitigation immediately if following a more aggressive protection policy.

Fig. 11. Examples of status graphs for explaining ICMP attacks.

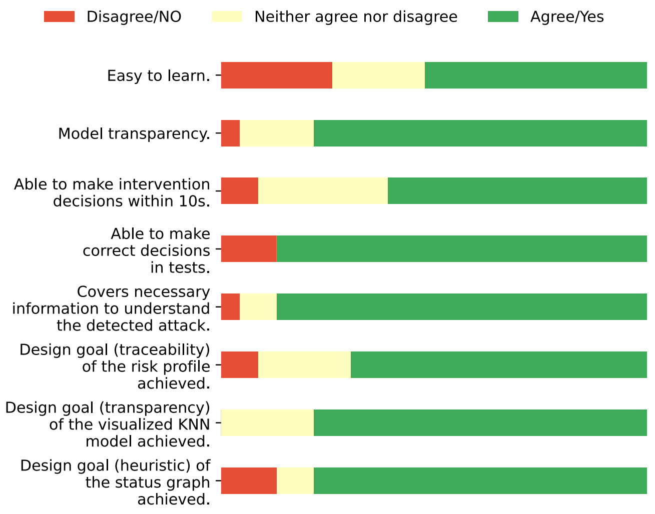


Fig. 12. Selected questionnaire evaluation results on the explainability of our approach.

1) The IP addresses in the network traffic data are anonymized in a prefix-preserving manner with CryptoPAN [60], before collection and storage. This ensures we cannot trace back to individual users during our deployment.
2) All the data is stored on a restricted server. Besides the SSH port, all the ports on the server are closed, and no connections can be initiated from the server. This minimizes the risk of accidental leakage of network data.
3) Our approach can only be deployed on the restricted server.
4) We are only allowed to receive the detection results from the restricted server. Other information related to the IXP operations have to remain on the restricted server, such as prefix-level measurements, the trained model, and pre-processed data.

2) Deployment Setup: The restricted server where our approach is deployed has an Intel Xeon Silver 4116 processor with 64 GB of RAM. The flow-level data is collected from multiple routers at FRGP during a 3-month period between 10:20 MST on February 24 to 21:40 MST on May 21, 2020. At its peak, the traffic volume usually reaches 100 Gbps during the day. Our approach can simultaneously obtain access to network traffic flows in three formats, which are NetFlow, Argus Flow [61], and FlowRide, a newly developed flow-capture tool that summarizes traffic every five seconds. The pre-processing module converts the traffic flows into the overall traffic profiles and IP-level traffic profiles for each detector.

As was true in the evaluation of the simulation environment, the deployed detection model was pre-trained with datasets shown in Table I. To adapt the pre-trained detection model to the FRGP environment, we conducted several measurements on the network to obtain the data distribution for each traffic feature used by the detectors. Then, we mapped the pre-trained model to the FRGP environment according to these distributions. While the program was running, we were able to receive the detection results for different types of attacks (i.e., NTP, TCP SYN, ICMP, and UDP attacks). During the evaluation, the FRGP operators also gave us information about DDoS attacks they discovered using Arbor Network's PeakFlow and Threat Mitigation System (TMS) [62]. Of a particular note is that the attacks reported by FRGP cannot represent ground truth as the IXP also suffers from false positives and false negatives, but they have good reference values for evaluating our approach. In addition, our contract with FRGP operators does not allow us to alter any traffic flows in their network, so we did not evaluate the classification efficacy in this deployment.

3) Findings: To better quantify the traffic change during an anomaly, we define **peak intensity index** ζ, calculated as $\zeta = V_{peak}/V_{exp}$, where V_{peak} denotes the peak volume of the anomaly and V_{exp} denotes the expected traffic volume. For an anomaly with a short duration (less than 30 minutes), we treat the traffic volume right before the anomaly as V_{exp}. For an anomaly with a longer duration, we calculate V_{exp} by statistically averaging legitimate traffic volumes at the same time in the surrounding seven days. Fig. 13 shows the anomaly detection results. The top subplot illustrates the peak intensity indexes ζ of the anomalies occurring at different times. The bottom subplot illustrates the duration of the detected anomalies at different times.

Our approach successfully detected over 90% of DDoS attacks reported by FRGP operators, including all severe attacks with a ζ greater than 2. The five missed alarms (highlighted with red circles in Fig. 13) were all low-intensity attacks that did not significantly damage the systems.

Furthermore, our approach proved more sensitive in detecting DDoS attacks, generating 21 alarms that FRGP operators missed (highlighted with yellow circles in Fig. 13). These 21 alarms involved low-intensity, short-duration attacks, which

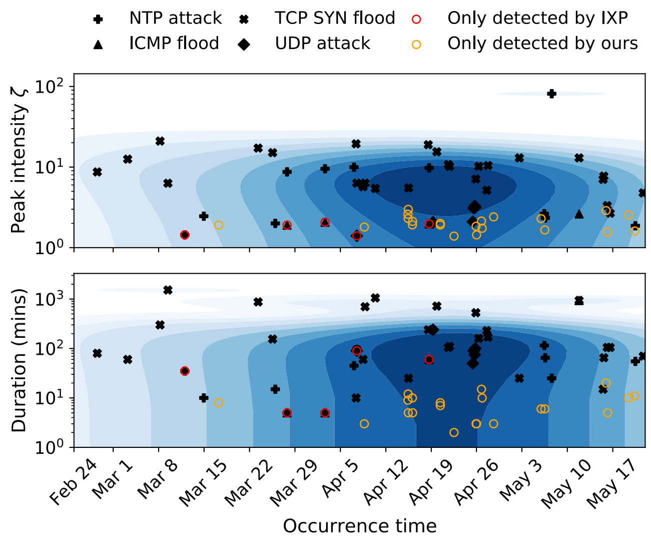


Fig. 13. Detection results from 10:20 MST on February 24 to 21:40 MST on May 21, 2020. Red circles indicate the attacks only reported by FRGP operators but not detected by our approach. Yellow circles indicate the attacks only detected by our approach but not reported by FRGP operators. Other dots indicate the attacks detected by both parties. The depth of the background color represents the density of attacks.

could represent small-scale floods undetected by FRGP's system or false positives. The effective explainability of our approach enabled network administrators to determine that most of these attacks were in preparatory or stalemate stages based on their status graphs. Consequently, if network administrators opt for a conservative mitigation policy, they can quickly review the explanatory information and choose to disregard these alarms.

In conclusion, the real-world deployment demonstrates the adaptability and usability of our approach. Besides, the explanatory information can quickly help the network administrators identify possible false positives or less threatening attacks, thereby making necessary interventions.

VI. Conclusion

This paper presents a learning-based approach for detecting DDoS traffic. In comparison to existing methods, the proposed approach offers two key advantages: (1) explainability and (2) adaptability. By employing a KD tree and a modified KNN algorithm, the method generates a tree-like classifier that not only accelerates predictions but also produces interpretable outputs. These outputs offer network administrators a clear understanding of network context, detection logic, attack stages, and mitigation cost-effectiveness. Additionally, users can easily adapt the detection model to different environments using prior knowledge, without the need to retrain the model from scratch. Leveraging grid sorting, the classification module significantly reduces collateral damage and delivers results promptly.

We trained the detection model using representative DDoS datasets from public repositories. We then evaluated the approach in both simulated and real-world settings. The evaluation results demonstrate the effectiveness and efficiency of this approach in both scenarios, as well as its adaptability from small simulated environments to a real IXP setting. Regarding explainability, the evaluation from our questionnaire illustrates that our approach successfully meets its design objectives in terms of transparency, traceability, heuristic principles, and ease of learning.

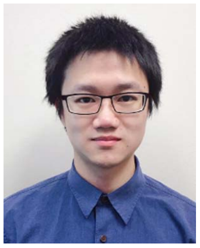
Acknowledgment

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Department of Homeland Security or the US Government.

References

[1] "National cyber awareness system: Security tip - understanding Denial-of-Service Attacks," 2009. [Online]. Available: https://www.us-cert.gov/ncas/tips/ST04--015
[2] "Cisco annual internet report (2018–2023) white paper," 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/executive-perspectives/annual-internet-report
[3] M. Suresh and R. Anitha, "Evaluating machine learning algorithms for detecting ddos attacks," in *Advances in Network Security and Applications*, D. C. Wyld, M. Wozniak, N. Chaki, N. Meghanathan, and D. Nagamalai, Eds. Berlin, Germany: Springer, 2011, pp. 441–452.
[4] X. Yuan, C. Li, and X. Li, "Deepdefense: Identifying DDoS attack via deep learning," in *Proc. IEEE Int. Conf. Smart Comput.*, 2017, pp. 1–8.
[5] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning DDoS detection for consumer Internet of Things devices," in *Proc. IEEE Secur. Privacy Workshops*, 2018, pp. 29–35.
[6] F. Emmert-Streib, O. Yli-Harja, and M. Dehmer, "Explainable artificial intelligence and machine learning: A reality rooted perspective," *Wiley Interdiscipl. Rev., Data Mining Knowl. Discov.*, vol. 10, no. 6, 2020, Art. no. e1368.
[7] M. Choraś, M. Pawlicki, D. Puchalski, and R. Kozik, "Machine learning–the results are not the only thing that matters! what about security, explainability and fairness?," in *Proc. 20th Int. Conf. Comput. Sci.*, Amsterdam, The Netherlands, Springer, Jun. 2020, pp. 615–628.
[8] F. K. Došilović, M. Brčić, and N. Hlupić, "Explainable artificial intelligence: A survey," in *Proc. IEEE 41st Int. Conv. Inf. Commun. Technol. Electron. Microelectronics*, 2018, pp. 0210–0215.
[9] W. Samek and K.-R. Müller, "Towards explainable artificial intelligence," in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Berlin, Germany: Springer, 2019, pp. 5–22.
[10] E. Tjoa and C. Guan, "A survey on explainable artificial intelligence (XAI): Toward medical XAI," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 11, pp. 4793–4813, Nov. 2021.
[11] The Front Range GigaPoP: Connecting colorado and wyoming with hundreds of gigabits, 2021. [Online]. Available: https://frgp.net/frgp/index.shtml
[12] M. H. Bhuyan, H. J. Kashyap, D. K. Bhattacharyya, and J. K. Kalita, "Detecting distributed denial of service attacks: Methods, tools and future directions," *Comput. J.*, vol. 57, no. 4, pp. 537–556, 2014.
[13] J. Mirkovic and P. Reiher, "D-WARD: A source-end defense against flooding denial-of-service attacks," *IEEE Trans. Dependable Secure Comput.*, vol. 2, no. 3, pp. 216–232, Third Quarter 2005.
[14] C.-L. Chen, "A new detection method for distributed denial-of-service attack traffic based on statistical test," *J. Universal Comput. Sci.*, vol. 15, no. 2, pp. 488–504, 2009.
[15] G. Zhang, S. Jiang, G. Wei, and Q. Guan, "A prediction-based detection algorithm against distributed denial-of-service attacks," in *Proc. Int. Conf. Wirel. Commun. Mobile Comput., Connecting World Wirelessly*, 2009, pp. 106–110.
[16] Kentik is network observability. Accessed: Jan. 11, 2021. [Online]. Available: https://www.kentik.com/
[17] R. Thomas, B. Mark, T. Johnson, and J. Croall, "NetBouncer: Client-legitimacy-based high-performance ddos filtering," in *Proc. IEEE DARPA Inf. Survivability Conf. Expo.*, 2003, pp. 14–25.
[18] J. Wang, R. C.-W. Phan, J. N. Whitley, and D. J. Parish, "Augmented attack tree modeling of distributed denial of services and tree based attack detection method," in *Proc. IEEE 10th Int. Conf. Comput. Inf. Technol.*, 2010, pp. 1009–1014.

[19] L. Limwiwatkul and A. Rungsawang, "Distributed denial of service detection using TCP/IP header and traffic measurement analysis," in *Proc. IEEE Int. Symp. Commun. Inf. Technol.*, 2004, pp. 605–610.
[20] Z. He, T. Zhang, and R. B. Lee, "Machine learning based DDoS attack detection from source side in cloud," in *Proc. 4th Int. Conf. Cyber Secur. Cloud Comput.*, 2017, pp. 114–120.
[21] M. Zekri, S. El Kafhali, N. Aboutabit, and Y. Saadi, "DDoS attack detection using machine learning techniques in cloud computing environments," in *Proc. 3rd Int. Conf. Cloud Comput. Technol. Appl.*, 2017, pp. 1–7.
[22] K. Lu, D. Wu, J. Fan, S. Todorovic, and A. Nucci, "Robust and efficient detection of DDoS attacks for large-scale internet," *Comput. Netw.*, vol. 51, no. 18, pp. 5036–5056, 2007.
[23] Y. Feng, J. Li, and T. Nguyen, "Application-layer DDoS defense with reinforcement learning," in *Proc. IEEE/ACM Int. Symp. Qual. Service*, 2020, pp. 1–10.
[24] J. Seo, C. Lee, T. Shon, K.-H. Cho, and J. Moon, "A new DDoS detection model using multiple SVMs and TRA," in *Proc. Int. Conf. Embedded Ubiquitous Comput.*, Springer, 2005, pp. 976–985.
[25] R. Kokila, S. T. Selvi, and K. Govindarajan, "DDoS detection and analysis in SDN-based environment using support vector machine classifier," in *Proc. IEEE 6th Int. Conf. Adv. Comput.*, 2014, pp. 205–210.
[26] M. Barati, A. Abdullah, N. I. Udzir, R. Mahmod, and N. Mustapha, "Distributed denial of service detection using hybrid machine learning technique," in *Proc. IEEE Int. Symp. Biometrics Secur. Technol.*, 2014, pp. 268–273.
[27] S. Mergendahl and J. Li, "Rapid: Robust and adaptive detection of distributed denial-of-service traffic from the Internet of Things," in *Proc. IEEE Conf. Commun. Netw. Secur.*, 2020, pp. 1–9.
[28] M. Wichtlhuber et al., "IXP scrubber: Learning from blackholing traffic for ML-driven ddos detection at scale," in *Proc. ACM SIGCOMM Conf.*, 2022, pp. 707–722.
[29] K. Lee, J. Kim, K. H. Kwon, Y. Han, and S. Kim, "DDoS attack detection method using cluster analysis," *Expert Syst. Appl.*, vol. 34, no. 3, pp. 1659–1665, 2008.
[30] L. Zi, J. Yearwood, and X.-W. Wu, "Adaptive clustering with feature ranking for DDoS attacks detection," in *Proc. IEEE 4th Int. Conf. Netw. Syst. Secur.*, 2010, pp. 281–286.
[31] W. Bhaya and M. EbadyManaa, "DDoS attack detection approach using an efficient cluster analysis in large data scale," in *Proc. IEEE Annu. Conf. New Trends Inf. Commun. Technol. Appl.*, 2017, pp. 168–173.
[32] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*, vol. 821. Hoboken, NJ, USA: Wiley, 2012.
[33] M. W. Gardner and S. Dorling, "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences," *Atmospheric Environ.*, vol. 32, no. 14/15, pp. 2627–2636, 1998.
[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.
[35] Y. Cao, H. Jiang, Y. Deng, J. Wu, P. Zhou, and W. Luo, "Detecting and mitigating DDoS attacks in SDN using spatial-temporal graph convolutional network," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 6, pp. 3855–3872, Nov./Dec. 2022.
[36] Q. P. Nguyen, K. W. Lim, D. M. Divakaran, K. H. Low, and M. C. Chan, "GEE: A gradient-based explainable variational autoencoder for network anomaly detection," in *Proc. IEEE Conf. Commun. Netw. Secur.*, 2019, pp. 91–99.
[37] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?" Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1135–1144.
[38] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Adv. Neural Inf. Process. Syst.*, vol. 30, pp. 4768–4777, 2017.
[39] R. Jalili, F. Imani-Mehr, M. Amini, and H. R. Shahriari, "Detection of distributed denial of service attacks using statistical pre-processor and unsupervised neural networks," in *Proc. Int. Conf. Inf. Secur. Pract. Experience*, 2005, pp. 192–203.
[40] P. A. R. Kumar and S. Selvakumar, "Distributed denial of service attack detection using an ensemble of neural classifier," *Comput. Commun.*, vol. 34, no. 11, pp. 1328–1341, 2011.
[41] S. N. Shiaeles, V. Katos, A. S. Karakos, and B. K. Papadopoulos, "Real time DDoS detection using fuzzy estimators," *Comput. Secur.*, vol. 31, no. 6, pp. 782–790, 2012.
[42] M. Zhang et al., "Poseidon: Mitigating volumetric DDoS attacks with programmable switches," in *Proc. 27th Netw. Distrib. Syst. Secur. Symp.*, 2020, pp. 1–18.
[43] C. Rossow, "Amplification hell: Revisiting network protocols for DDoS abuse," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2014, pp. 1–15.
[44] A. Praseed and P. S. Thilagam, "DDoS attacks at the application layer: Challenges and research perspectives for safeguarding web applications," *IEEE Commun. Surv. Tut.*, vol. 21, no. 1, pp. 661–685, First Quarter 2019.
[45] B. Claise, "Cisco systems netflow services export version 9," RFC 3954, IETF, Oct. 2004, doi: 10.17487/RFC3954.
[46] S. Panchen, N. McKee, and P. Phaal, "InMon corporation's sFlow: A method for monitoring traffic in switched and routed networks," Sep. 2001, doi: 10.17487/RFC3176.
[47] M.-L. Zhang and Z.-H. Zhou, "ML-KNN: A lazy learning approach to multi-label learning," *Pattern Recognit.*, vol. 40, no. 7, pp. 2038–2048, 2007.
[48] J. H. Friedman, "On bias, variance, 0/1—loss, and the curse-of-dimensionality," *Data Mining Knowl. Discov.*, vol. 1, no. 1, pp. 55–77, 1997.
[49] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," *VISAPP (1)*, vol. 2, no. 331/340, 2009, Art. no. 2.
[50] K. Li, W. Zhou, P. Li, J. Hai, and J. Liu, "Distinguishing DDoS attacks from flash crowds using probability metrics," in *Proc. 3rd Int. Conf. Netw. Syst. Secur.*, 2009, pp. 9–17.
[51] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics Intell. Lab. Syst.*, vol. 2, no. 1/3, pp. 37–52, 1987.
[52] DARPA 2009 intrusion detection dataset (colorado state university), 2009. [Online]. Available: http://www.darpa2009.netsec.colostate.edu/
[53] The CAIDA UCSD "DDoS attack 2007" dataset, 2007. [Online]. Available: https://www.caida.org/data/passive/ddos-20070804_dataset.xml
[54] FRGP NTP flow data - NTP reflection attack, 2014. [Online]. Available: https://www.impactcybertrust.org/dataset_view?idDataset=776
[55] DDoS Chargen 2016 dataset - Internet traffic data containing a DDoS attack based on UDP Chargen protocol, 2016. [Online]. Available: https://www.impactcybertrust.org/dataset_view?idDataset=693
[56] W. S. Noble, "What is a support vector machine?," *Nat. Biotechnol.*, vol. 24, no. 12, pp. 1565–1567, 2006.
[57] P. Odintsov, "FastNetMon–very fast DDoS analyzer with sflow/netflow/mirror support,", 2020. [Online]. Available: https://github.com/pavel-odintsov/fastnetmon/
[58] MiniNet: An instant virtual network on your laptop (or other PC). Accessed: Dec. 11, 2020. [Online]. Available: http://mininet.org/
[59] X. Wang and M. Yin, "Are explanations helpful? A comparative study of the effects of explanations in AI-assisted decision-making," in *Proc. 26th Int. Conf. Intell. User Interfaces*, 2021, pp. 318–328.
[60] J. Xu, J. Fan, M. Ammar, and S. B. Moon, "On the design and performance of prefix-preserving IP traffic trace anonymization," in *Proc. 1st ACM SIGCOMM Workshop Internet Meas.*, 2001, pp. 263–266.
[61] Argus: The first network flow system, 2019. [Online]. Available: https://openargus.org/
[62] Arbor threat mitigation system. Accessed: Jan. 11, 2021. [Online]. Available: https://www.netscout.com/product/arbor-threat-mitigation-system



Yebo Feng received the BS degree in computer science from Yangzhou University in 2016, the ME degree in computer science from the University of Oregon (UO) in 2018, and the PhD degree from the University of Oregon (UO) in 2023. His research interests include network security, blockchain security, DDoS defense, and network traffic analysis. He is the recipient of the Best Paper Award of 2019 IEEE CNS, Gurdeep Pall Graduate Student Fellowship of UO, and Ripple Research Fellowship. He has served as the reviewer of *IEEE Transactions on Dependable and Secure Computing*, *ACM Transactions on Knowledge Discovery from Data*, and *IEEE Journal on Selected Areas in Communications*. He was also on the program committees of several international conferences, such as CYBER, SECURWARE, and B2C.

Jun Li (Senior Member, IEEE) received the BS degree in computer science from Peking University, the ME degree in computer science from the Chinese Academy of Sciences (with a Presidential Scholarship), and the PhD degree in computer science from UCLA (with the distinction of Outstanding Doctor of Philosophy). He is a professor with the Department of Computer Science and director of the Network & Security Research Laboratory with the University of Oregon. He was also a Ripple fellow, a Narus research fellow, and the founding director of the Center for Cyber Security and Privacy with the University of Oregon. His research focuses on networking, distributed systems, and network security. He has published more than 100 peer-reviewed papers, including several best paper awards. He has received the CAREER Award from the US National Science Foundation, the Faculty Excellence Award from the University of Oregon, and the Recognition of Service Award from ACM, among many others. He has served on US National Science Foundation research panels, editorial boards of networking and security journals, and more than 70 international technical program committees, including serving on the steering committees of several and chairing a few.

Devkishen Sisodia received the BS degree in computer science from the University of Texas at Arlington and the PhD degree from the Department of Computer and Information Science, the University of Oregon, where he conducted research with the Center for Cyber Security and Privacy. He is an Assistant Professor with the Department of Computer Science & Software Engineering, California Polytechnic State University, San Luis Obispo. His primary area of research revolves around network security, with a particular focus on distributed denial-of-service (DDoS) attacks and defenses, as well as Internet of Things (IoT) security and privacy. His other research interests include Internet measurement and cybersecurity education.

Peter Reiher (Member, IEEE) received the PhD degree from UCLA in 1987. He is an adjunct professor with the Computer Science Department, UCLA. His research interests include cybersecurity (particularly Internet security issues, such as distributed denial of service attacks), computer networks, ubiquitous computing, security of autonomous vehicles and vehicular networks, security of wireless medical devices, and file systems.