DDoSHIELD-IoT: A Testbed for Simulating and Lightweight Detection of IoT Botnet DDoS Attacks

Simona De Vivo*, Islam Obaidat†, Dong Dai†, and Pietro Liguori*
*University of Naples Federico II, Naples, Italy
{simona.devivo, pietro.liguori}@unina.it
†University of North Carolina at Charlotte, NC, USA
{iobaidat, ddai}@charlotte.edu

***Abstract*—In the rapidly expanding realm of the *Internet of Things* (IoT), the escalation of sophisticated cyber threats, particularly botnet *Distributed Denial of Service* (DDoS) attacks, highlights the importance of *Intrusion Detection Systems* (IDS) for maintaining network integrity. IDSs are necessary tools for identifying and mitigating such threats. Consequently, there is a compelling need for a testbed that can facilitate the development and rigorous evaluation of IDS solutions, specifically designed to meet unique requirements and constraints of IoT environments. To bridge this gap, DDoSHIELD-IoT, an IDS testbed, is introduced, aiming to provide a platform for creating and evaluating IDSs within the IoT context. DDoSHIELD-IoT leverages Docker containers and the NS-3 network simulator to accurately mimic IoT environments and traffic. DDoSHIELD-IoT is used to implement and evaluate multiple IDSs. These IDSs leverage different machine learning models, such as K-Means, to detect Mirai botnet DDoS traffic, achieving an accuracy of over 90%. This evaluation highlights DDoSHIELD-IoT's precision as an IDS testbed. Furthermore, DDoSHIELD-IoT provides the capability to measure diverse performance metrics, such as CPU and memory usage. These assessments show DDoSHIELD-IoT's contributions to IoT security practices by offering scalability and reproducibility for enhanced IDS creation and evaluation.**

***Index Terms*—IDS, Botnet, DDoS, IoT**

I. INTRODUCTION

In recent years, *Internet of Things* (IoT) has witnessed widespread adoption, which benefits various sectors, such as smart homes, smart cities, healthcare, and autonomous vehicles [1]. IoT pervasiveness leads to a notable surge in connected devices, thus widening the potential attack surface [2]. Consequently, the IoT environment has become the primary target for cyber-attacks in recent years, particularly botnet *Distributed Denial of Service* (DDoS) attacks [3], owing to its increased vulnerability and expanded network connectivity [4].

Given the imperative need to fortify our networks against these threats, the integration of robust *Intrusion Detection System* (IDS), tailored for IoT environments, becomes essential [5]. These IDSs must consider the limited resources of IoT devices and the characteristics of IoT environments [6]. Therefore, we need an environment that assists IoT IDS research [7]. Several IoT IDSs have been suggested in the literature [8]–[18], that present testbed for evaluating DDoS attack detection algorithms and simulating IIoT networks.

However, these approaches have limitations in performance assessment, arising from inadequate evaluation metrics, non-representative test environments, unrealistic test data, lack of diversity in simulated attacks, poor reproducibility, and subjective interpretation of results, hindering a comprehensive understanding of IDS effectiveness in IoT environments [19], [20]. Moreover, the IDS-testbed implementation presents further limitations related to practical implementation. Indeed, implementing a testbed replicating the real industrial environments is complex due to the variety and complexity of devices and protocols used in such contexts. Similarly, replicating realistic attack scenarios in home contexts is challenging due to the diversity of home devices and networks, potential threats variety, and anomalous behavior [21], [22].

This paper presents DDoSHIELD-IoT, a simulation testbed that uses Docker containers and the NS3 simulator to facilitate IoT IDS research. This setup enables DDoSHIELD-IoT to support the deployment and operation of IDSs housed within Docker containers alongside IoT binaries. In addition, the NS-3 network simulator allows the generation and capturing of real-world benign and malicious traffic within the simulated network. This captured traffic can be used as datasets to train and develop robust IDSs tailored for IoT networks, addressing the lack of high-quality datasets required to build IoT IDSs.

DDoSHIELD-IoT is constructed upon the foundation provided by `DDoSim`, a simulation tool specifically designed for studying DDoS attacks in the IoT domain [18]; `DDoSim`'s primary objective is to mimic the behavior of botnets and DDoS attacks in a controlled environment. Building upon `DDoSim`, DDoSHIELD-IoT extends its capabilities by incorporating real-world benign and malicious traffic into the framework—an enhancement that enriches the realism of the testbed and provides a more comprehensive platform for the development, testing, and evaluation of IoT-specific IDSs. In particular, DDoSHIELD-IoT generates two categories of traffic: benign, consisting of file transfer data (FTP traffic), video streaming data (RTMP traffic), and HTTP traffic, and malicious, represented by data obtained from the Mirai malware [23], [24].

We test and evaluate DDoSHIELD-IoT by implementing multiple real-time IDSs designed to detect the Mirai DDoS traffic. We implement these IDSs using different machine learning models, including *Random Forest* (RF), K-Means, and *Convolutional Neural Network* (CNN). These implementations allow us to evaluate and compare these different IDSs within the same environment in terms of performance (e.g., detection

actuary) and resource consumption (e.g., CPU and Memory). Our experimental results show that these IDSs achieve a high detection accuracy (95.47%) in realistic settings.

In addition to assessing IDS effectiveness based on traditional metrics, DDoSHIELD-IoT offers a comprehensive approach to evaluating IoT-specific IDS solutions. Recognizing the importance of scalability, adaptability, and robustness, DDoSHIELD-IoT provides a customizable environment leveraging Docker containers, allowing researchers to modify and extend the framework to suit their specific needs. Furthermore, we promote open research practices by sharing DDoSHIELD-IoT's entire source code with the research community, facilitating collaboration and reproducibility. By offering a modular and extensible architecture, DDoSHIELD-IoT enables the seamless integration of new detection algorithms, simulation scenarios, and evaluation methodologies. Through these initiatives, we aim to foster collaboration, innovation, and advancement in IoT security research.

In summary, this paper presents the following contributions:

- We present DDoSHIELD-IoT, a testbed that hosts actual IoT binaries, which communicate over a simulated network; DDoSHIELD-IoT can generate real-world IoT traffic and implement (and evaluate) IoT IDSs to detect botnet DDoS attacks.
- We use DDoSHIELD-IoT to implement and evaluate multiple IDSs based on different ML models (such as K-Means and CNN).
- We publicly release DDoSHIELD-IoT at https://github.com/iobaidat/DDoShield-IoT for the research community to reproduce our experiments and to test their own IDS implementations.

The rest of this paper is organized as follows. Section II discusses the related work; Section III describes the proposed framework; Section IV presents our evaluation of DDoSHIELD-IoT; Section V discusses the threats to validity; Section VI concludes our paper.

II. Related Work

In the IoT networks intrusion detection domain, various approaches have been proposed [8]–[18]. For instance, Bhayo et al. [8] propose a testbed aimed at evaluating DDoS attack detection algorithms in SD-Networks for IoT environments. Through this testbed, the authors simulate SDN-IoT traffic and conduct detailed parameter analysis. They employ the SDN-WISE environment and WEKA for classification purposes. Zolanvari et al. [9] introduce a testbed for studying Industrial Internet of Things (IIoT) networks using machine learning, which simulates a real industrial plant to monitor water level and turbidity in a tank. Their implementation involves both hardware and software components to collect realistic data, including normal traffic and cyber attacks, for evaluating the effectiveness of machine learning models in threat detection. Anthi et al. [10] propose a testbed simulating a smart home IoT environment, incorporating various IoT devices, to evaluate an intrusion detection system (IDS) tailored for domestic IoT devices. Nie et al. [11] present a testbed designed to simulate an IoV scenario with DDoS attacks, aiming to realistically replicate traffic flows within an IoV and simulate various attack scenarios using TCP, UDP, and HTTP crowding techniques. Albulayhi et al. [14] propose a testbed simulating a wide range of real attacks in IoT environments using devices such as AI speakers, Wi-Fi cameras, and laptops. Its novelty lies in the ability to simulate various attack scenarios and involve real hardware components to create a realistic IoT environment. Kumar et al. [15] introduce the "EDIMA" testbed, which aims to detect malware activity in IoT networks using machine learning techniques. By integrating network traffic sensors, a data collection server, and monitoring systems, it can capture a wide range of suspicious behaviors. Zachos et al. [16] introduce the IoT/IoMT Security Testbed, which simulates sensor functionality using Raspberry Pi devices with Raspbian OS. Passban IDS: An Intelligent Anomaly-Based Intrusion Detection System for IoT Edge Devices, proposed by Eskandari et al. [17], introduces a simulated environment for evaluating Passban, an intrusion detection system, in IoT networks. Finally, Obaidat et al. [18] present `DDoSim`, a framework specifically designed for simulating and assessing large-scale botnet DDoS attacks. While emphasizing scalability, cost-efficiency, and real-time analysis, `DDoSim` lacks real-world benign traffic generation and an IDS component for ML-based intrusion detection models evaluation and comparison.

While these works provide valuable contributions to IoT intrusion detection research, they exhibit certain limitations. Some testbeds, for instance, do not offer a comprehensive evaluation of security system performance, thus limiting understanding of their effectiveness in realistic scenarios. Moreover, fully replicating real IoT network environments can be complex and resource-intensive, limiting testbeds' ability to simulate diverse and complex IoT environments. Additionally, some testbeds may not simulate all possible attack and traffic scenarios occurring in real IoT networks, thereby affecting their ability to provide a complete representation of IoT network security.

DDoSHIELD-IoTaims to address these issues by providing an advanced and comprehensive simulation environment for IoT IDS research. Firstly, DDoShield-IoT integrates Docker containers and the NS3 network simulator to facilitate IDS implementation and operation within Docker containers, offering greater flexibility and ease in managing the simulation environment. This feature provides a more advanced and streamlined implementation compared to other testbeds that may require more complex configurations. Furthermore, DDoShield-IoT extends the capabilities of the `DDoSim` simulator, used for studying DDoS attacks in the IoT domain, by incorporating real benign and malicious traffic into the framework. This addition significantly enhances the realism of the testbed, providing a more comprehensive and representative platform for developing, testing, and evaluating IoT-specific IDSs. Another innovation of DDoShield-IoT is the ability to generate and evaluate multiple real-time IDSs using machine learning models such as Random Forest, K-Means, and Convolutional Neural Network (CNN). This capability allows for a more

in-depth comparative evaluation of IDS performance in the same simulation environment, offering users greater flexibility in choosing the model best suited to their needs. Lastly, DDoShield-IoT distinguishes itself through its commitment to sharing and collaboration within the research community, offering the public release of the framework to enable experiment reproducibility and provide other researchers with the opportunity to test and validate their IDS implementations.

These approaches contribute to the advancement of intrusion detection systems and the evaluation of IoT network security in various contexts. However, they also highlight significant limitations, underscoring the need for a novel testbed for IDS evaluation. Current challenges encompass complex implementation, context, data representativeness, scalability and adaptability, and performance assessment of IDS.

Our framework DDoSHIELD-IoT aims to address the shortcomings of existing approaches through simplified usability, enhanced realism, scalability and adaptability, comprehensive evaluation metrics, and the integration of advanced technologies like machine learning. This novel approach offers an effective solution for IDS evaluation in IoT networks, providing a controlled and realistic environment to test and compare different defense strategies against evolving cyber threats.

III. The DDoSHIELD-IoT Framework

In this section, we introduce DDoSHIELD-IoT, an IDS simulation testbed derived from the pre-existing `DDoSim` framework [18]. We have retained the core features of `DDoSim` while enhancing its functionality to cater to a distinct use case: evaluating IDS performance within IoT systems.

A. DDoSim Overview

`DDoSim` [18] is a simulation environment designed for replicating large-scale botnet DDoS attacks in a realistic scenario. It allows for modeling all phases of the attack, enabling, through external tools, e.g., Wireshark[1], real-time analysis at various stages, e.g., the severity of the impact, such as alterations in the target server's throughput, the average data reception frequency, and the number of connected bots. The communication in `DDoSim` is facilitated through a customizable simulated network, making it adaptable to different use cases, including CSMA and Wi-Fi networks. This framework integrates Docker[2], an open-source platform for creating, deploying, and managing applications in isolated environments known as containers, and NS-3[3], an open-source simulation framework for communication networks. Leveraging the dynamic capabilities of Docker containers, `DDoSim` emulates the main components of a typical botnet DDoS attack scenario: the Attacker, Devs, and TServer.

The Attacker component, encapsulated within a Docker container node, orchestrates the assault by exploiting vulnerabilities within the Devs and deploying botnet malware. Employing a suite of meticulously crafted Exploit & Infection Scripts tailored to specific device vulnerabilities, the Attacker infiltrates and compromises the Devs. This infiltration triggers the installation and execution of botnet malware, establishing a foothold for command and control operations. The Command and Control Server, a pivotal subcomponent, serves as the central hub for coordinating the activities of the compromised Devs, issuing orchestrated commands for launching botnet DDoS attacks against the designated target server, TServer.

Devs, representing a diverse array of network-facing devices, are provisioned as Docker containers housing vulnerable binaries. These containers simulate the susceptibility of real-world devices to cyber threats, serving as unwitting foot soldiers in the botnet army. TServer, the ultimate target of the assault, is a simulated node within the NS-3 network, engineered to withstand the barrage of malicious traffic unleashed by the compromised Devs.

Underpinning `DDoSim`'s functionality is a crafted network configuration implemented through Docker containers and NS-3 ghost nodes combination. Each Docker container, representing either the Attacker or a Dev, interfaces with the NS-3 simulated network through specialized bridge configurations. This intricate network topology enables seamless communication and data exchange between the simulated components, faithfully replicating real-world network dynamics.

Moreover, `DDoSim` facilitates comprehensive experimentation through designed scenarios, crafting experiments to examine several factors, such as churn rates and attack duration. For instance, by varying churn rates, `DDoSim` enables the assessment of the impact of device mobility and connectivity on the resilience of TServer to botnet DDoS attacks. Similarly, manipulating attack duration makes it possible to gauge the temporal evolution of attack intensities and devise proactive defense strategies accordingly. The results obtained from these experiments serve as invaluable benchmarks for evaluating the effectiveness of defense mechanisms, ranging from intrusion detection systems to traffic filtering and mitigation techniques.

Furthermore, `DDoSim`'s validation procedures attest to its robustness and fidelity in simulating real-world cyber threats. Through rigorous comparison with actual hardware experiments, `DDoSim` validates the accuracy and reliability of its simulated outcomes. By replicating experimental conditions and configurations, `DDoSim` ensures that its simulations closely mirror the behavior of physical network environments.

B. DDoShield-IoT's Improvements

As shown in Figure 1, our framework improves `DDoSim`, enhancing the TServer and introducing a fourth Docker container simulating an IDS capable of real-time detection of botnet DDoS attacks executed by Devs against the TServer. Particularly:

- The TServer in DDoSHIELD-IoT includes three main servers: Apache, Nginx, and a customized FTP-Server. These servers generate real-world benign traffic, i.e., HTTP traffic, video traffic, and FTP traffic, respectively. A diverse set of regular activities in the simulation

[1]https://www.wireshark.org/
[2]https://www.docker.com/
[3]https://www.nsnam.org/

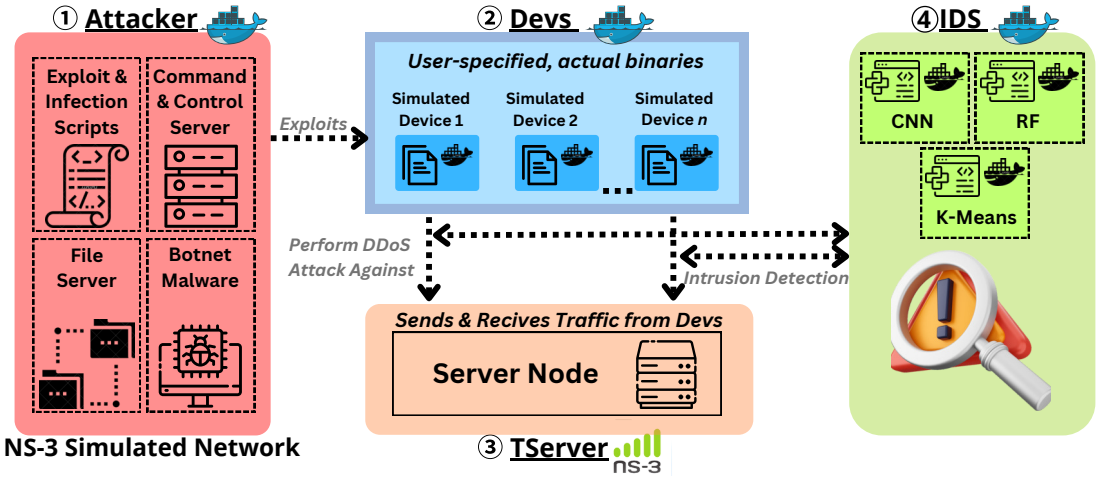

Fig. 1. DDoSHIELD-IoT Overview

environment is crucial to avoid an unbalanced dataset and enable systems to adapt to complex scenarios. The benign traffic generated by the TServer helps intrusion detection algorithms recognize proper traffic patterns by establishing a baseline for anomaly detection. Benign traffic in the dataset also assesses algorithm adaptability to dynamic network activity changes and helps prevent false positives by allowing systems to differentiate between legitimate and potentially harmful behaviors. Incorporating benign and malicious traffic in botnet DDoS attack simulation presents a more realistic challenge and enhances the effectiveness and robustness of ML security solutions.

- The Real-Time IDS Unit leverages ML models for botnet DDoS attack detection. Notably, based on user needs, it performs anomaly traffic detection using one of the following algorithms: *Convolutional Neural Network* (CNN), *Random Forest* (RF), and K-Means.

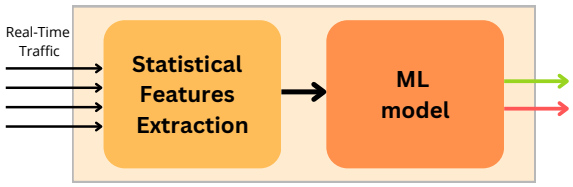

Fig. 2. IDS Component Overview

As illustrated in Figure 2, the IDS operation comprises three stages: i) real-time monitoring traffic, ii) preprocessing of data, and iii) identifying the attack.

The network traffic analysis starts with the features extraction process consisting of aggregating the incoming features for a time window defined by the users according to their needs (e.g., in our experiments, we chose a time window of 1 second). This process calculates the statistical features, which are equal for each packet in the time window. Then, it adds the statistical features to the basic features. Basic features aggregation with statical ones is indispensable to prevent the misclassification of packets belonging to different classes (malicious or benign) within the same time window. This aggregation provides an enhancement in the overall accuracy. Once the preprocessing data phase ends, the ML model, previously trained, executes the intrusion detection activity on the new data. The ML models provide a traffic classification as benign or malicious, also reporting the accuracy for each intrusion detection activity in the specific time window.

IV. Experimental Evaluation

This section evaluates the validity and versatility of the DDoSHIELD-IoT framework as an IDS testbed. Our evaluation includes various aspects, including data preparation, feature extraction, evaluation metrics, and resource utilization analysis. By evaluating these components, we aim to assess the capabilities of the DDoSHIELD-IoT as a comprehensive IDSs' experimentation and evaluation platform.

We conduct our experiment using a laptop with 16 GB memory and a 2.7 GHz Intel Core i5 CPU. We run DDoSHIELD-IoT on a virtual machine with Ubuntu 22.04 LTS as the guest OS. We implement DDoSHIELD-IoT with Docker version 20.10 and NS-3 simulator version 3.38.

A. Data Processing

The feature extraction process aims to reduce the number of input variables used in developing a predictive model, filtering out irrelevant variables and selecting only those most useful for accurate predictions [25]. This process plays a pivotal role in our study, serving as a critical step in the network traffic analysis for intrusion detection. As in [26]–[28], we do a manual feature extraction [4], selecting only the relevant features, particularly useful in identifying abnormal network activity, such as botnet DDoS attacks.

[4]Feature selection algorithms, such as *Recursive Feature Elimination* (RFE) [29], are not used in our study since we want to showcase our IDS testbed's capabilities, and thus, feature selection is beyond the scope of our work.

Our feature extraction process begins by capturing a comprehensive array of attributes from network packets, such as timestamps, IP source and destination addresses, protocol types (e.g., TCP, UDP), and source and destination ports. These attributes lay the groundwork for understanding the basic characteristics of network communication.

To go deeper in our analysis, we integrate several statistical features to uncover subtle patterns and anomalies within the network traffic. One crucial aspect is the calculation of packet counts within specified time windows, providing insights into traffic volume and fluctuations over time. This allows us to discern between normal traffic patterns characterized by consistent packet counts and irregularities indicative of potential attacks, such as sudden spikes or drops in packet counts. Additionally, we employ entropy analysis of destination ports to gauge the diversity and randomness of port usage. Normal scenarios often exhibit a wide range of destination ports with relatively uniform distribution, reflecting diverse communication activities. In contrast, attack scenarios may display skewed or concentrated distribution patterns, suggesting targeted or malicious port usage. Such deviations from normal port entropy values can indicate potential scanning or probing activities associated with malicious behavior.

Further enhancing our intrusion detection capabilities, we conduct frequency analysis of specific port usage, allowing us to identify trends and anomalies in port communication patterns. For instance, short-lived connection identification and repeated connection attempt detection are instrumental in spotting aggressive or suspicious behavior indicative of various attacks, including DDoS, since they differ from the normal traffic behavior, which shows stable and predictable port usage patterns, characterized by consistent frequencies of communication across various ports.

The feature extraction process also includes an in-depth examination of network scanning activity, focusing on SYN flags without corresponding ACK flags presence, which often precede aggressive attacks. By analyzing these and other statistical features, such as flow rates and sequence number variances, we gain a nuanced understanding of network dynamics, effectively discerning between benign and malicious activities.

By aggregating basic and statistical features, the IDS within DDoSHIELD-IoT has a detailed representation of network behavior. This multi-faceted feature extraction process lays the foundation for subsequent stages of the intrusion detection process, helping to provide the IDS with accuracy and robustness in distinguishing between benign and malicious activity.

B. Models

The algorithms employed in the IDS component of DDoSHIELD-IoT, are Random Forest, K-Means, and Convolutional Neural Network. These three distinct solutions represent different intrusion detection techniques with different features and benefits. Particularly:

- **Convolutional Neural Network (CNN)**: CNN is a type of neural network inspired by visual perception, capable of automatically extracting features from data using convolutional structures. Unlike traditional methods, CNNs do not require manual feature extraction and offer advantages such as local connections, weight sharing, and downsampling dimension reduction. Key components of a CNN model include convolution, padding, stride, and pooling, with dilated convolution allowing for the perception of larger areas without increasing parameters. CNNs are fundamental in deep learning, providing efficient feature extraction for tasks like image recognition and classification [30].
- **K-means**: The K-means model presented in [31] introduces a novel approach to tackle challenges commonly encountered in traditional clustering algorithms. This model dynamically determines the optimal number of clusters by incorporating entropy-based penalty terms into its objective function, resulting in enhanced clustering performance. The core functionality of the K-means algorithm revolves around iteratively updating cluster assignments, cluster centers, and mixing proportions. Initially, it randomly assigns data points to clusters and computes the centroids of these clusters. Then, it iteratively adjusts the cluster assignments based on the proximity of data points to the centroids, updates the centroids to the mean of the points in each cluster, and recalculates the mixing proportions. This iterative process continues until convergence, where the cluster assignments stabilize and centroids no longer shift significantly. The model's adaptability and stability during clustering iterations contribute to its simplicity and computational efficiency, rendering it particularly promising for unsupervised clustering tasks in IoT environments.
- **Random Forest (RF)**: The RF model is a robust and versatile machine learning algorithm widely used for classification and regression tasks. It creates an ensemble of decision trees, each trained on a random subset of the training data and features. This randomness is introduced through two key mechanisms: random sampling of training data using bagging and random selection of features for each node split in the decision trees. By leveraging the diversity among the trees, Random Forests reduce overfitting and improve generalization performance. During prediction, the ensemble of trees collectively votes or averages to produce the final output, offering robustness against noisy data and outliers. The parallelizable nature of decision tree construction also contributes to the efficiency of Random Forest training on large datasets [32].

To implement the RF and K-means models, we employ Scikit-Learn [5], a Python-based open-source machine-learning library, that offers a variety of algorithms and tools to create and evaluate machine-learning models. Instead, for CNN implementation, we use TensorFlow [6], an open-source library

[5]https://scikit-learn.org/stable/

[6]https://www.tensorflow.org/learn?hl=it

developed by Google for numerical computation and machine learning. It is widely used to create and train deep learning models, such as artificial neural networks, due to its flexibility and optimized performance for both CPU and GPU.

C. Evaluation Metrics

To assess the effectiveness of the attack detection models in identifying threats, we utilize four widely recognized anomaly detection evaluation metrics, including Precision, Recall, F1-Score, and Accuracy, collectively gauge a predictive model's performance by assessing its ability to make accurate predictions, identify relevant positive cases, strike a balance between Precision and Recall, and provide an overall assessment of correctness in predictions.

The computational resources available in IoT devices can vary significantly from one device to another. Generally, IoT devices have limited processing power, memory, and energy supply, which creates specific constraints for machine learning algorithms designed for IoT applications [33]. Therefore, it is crucial to optimize algorithms to ensure compatibility with IoT devices and to enable efficient operation within these constraints [34].

In this work, over the models' accuracy, we also assessed their sustainability and their compatibility with IoT devices, calculating the CPU usage percentage (%), the occupied RAM in *Kilobytes* (Kb), and the Model size in Kb. These metrics provide valuable insights into the resource consumption of machine learning models on IoT devices [35].

Using resource-aware ML models is significant in achieving cost savings and device longer lifecycles. Resource-aware ML models also contribute to reducing the energy footprint of IoT deployments. So, by calculating these parameters, we can also evaluate the environmental and economic sustainability of ML applications in IoT contexts, ultimately guiding the development of more efficient and eco-friendly solutions [36].

D. Performance Evaluation

To evaluate the performance of our IDS within the DDoSHIELD-IoT environment, we conducted a series of experiments focusing on training the models and real-time detection. Initially, we ran DDoSHIELD-IoT for 10 minutes to generate a dataset encompassing benign and malicious traffic. We use this dataset for model training, which is essential for subsequent real-time detection tasks. Particularly, the TServer generates benign traffic, as described in §III, while we generate malicious traffic using the Mirai malware, which can execute several botnet DDoS attacks [18]. The chosen attacks include SYN Flood, overwhelming servers by inundating them with SYN requests and depleting resources, ACK Flood, flooding systems with numerous ACK packets to disrupt regular network operations, and UDP Flood, which inundates servers with a high volume of UDP packets, leading to network congestion and potential service disruption. These specific attacks are selected to demonstrate the basic functionality of the IDS within the DDoSHIELD-IoT framework, avoiding more complex application-level attacks like HTTP Flood or DNS Flood, which necessitate additional application-level analysis. The resulting dataset is nearly balanced, comprising 3,012,885 malicious packets and 2,243,634 benign packets.

The IDS uses the RF, K-Means, and CNN models. We train these models using the training dataset mentioned above. We conduct training and evaluate the performance of these models using key evaluation metrics, including accuracy, precision, recall, and F1-score. Remarkably, all models have attained values across these evaluation metrics, with a small amount of false positives and false negatives. After the training phase, we save each model in a PKL file[7]. Saving the model after the training phase involves persisting its learned parameters, enabling later use without retraining. Indeed, we use these saved models for real-time detection.

We run the DDoSHIELD-IoT for 5 minutes for the real-time intrusion detection process. Throughout this duration, we extract statistical features from packets collected within a 1-second time window, assessing the accuracy of the models for that specific period. It is important to emphasize that users can customize the duration of the time window according to their analysis requirements.

Upon completion of the simulation time, we compute the average accuracy for each ML model. Unlike the training phase, where various evaluation metrics are considered, during the real-time detection, we solely calculate accuracy. This approach is required due to the characteristics of the simulation environment. Although benign and malicious traffic are generated simultaneously, there are instances during the real-time detection process when only one type of traffic is detected. For example, there may be intervals where only malicious traffic is identified, followed by periods with benign traffic only. Consequently, calculating precision and recall metrics during these isolated periods may result in division by zero, rendering these metrics unreliable. Hence, focusing exclusively on accuracy ensures a consistent and meaningful evaluation, even in scenarios with only one type of traffic.

Table I shows the average accuracy for each ML model at the end of the simulation time.

TABLE I
ML MODELS PERFORMANCE EVALUATION IN REAL-TIME DETECTION.

Model	Accuracy (%)
RF	61.22
K-Means	94.82
CNN	**95.47**

Analyzing the accuracy score related to each second during the simulation, we notice that the first and the last second of an attack duration report a drop in the model accuracy. The minimum registered is 35% for the K-Means model. This is due to the use of the statistical features that make noise since they are equal for each packet collected in that specific second. This happens because we do not use a features extraction

[7]A PKL, or Pickle file, is a serialized file format used in Python to store objects, including data structures, models, or any Python object.

algorithm that evaluates the actual usefulness of each feature after the basic and statistical features aggregation. This will be part of future work.

E. Sustainability Evaluation

Table II, shows CPU usage (%), the occupied RAM (Kb), and the model size (Kb) evaluated during the real-time detection process.

TABLE II
ML MODELS SUSTAINABILITY.

Model	CPU (%)	Memory (Kb)	Model Size (Kb)
RF	**65.46**	98.07	712.30
K-Means	67.88	**86.83**	**11.20**
CNN	65.94	275.85	736.30

The sustainability analysis highlights elevated CPU usage, resulting primarily from the IDS's computation of statistical features every second. A strategic approach to mitigate this high CPU usage involves adjusting the frequency at which statistical features are computed. By extending the period for computing these features, a reduction in CPU utilization can be achieved.

Furthermore, monitoring the allocated RAM and considering the model size suggests that the deep learning model, particularly CNN, is anticipated to have the most substantial size. In contrast, the k-Means model is the lightest concerning resource consumption.

V. THREATS TO VALIDITY

In evaluating the DDOSHIELD-IoT testbed, several potential threats to validity must be considered. These threats could affect the generalizability of our results and the applicability of the testbed in real-world scenarios.

An important consideration regarding the validity of our study is the limited representativeness of benign traffic considered. Although our framework offers a generic approach, its effectiveness is closely related to the quality and diversity of traffic data used for analysis. Currently, we only analyzed three types of protocols for benign traffic, i.e., HTTP, video, and FTP traffic. However, this selection may not be exhaustive, considering the wide range of protocols used in the IoT environment, characterized by considerable heterogeneity. This limitation may affect the ability of our framework to accurately detect and mitigate threats present in more complex and diverse scenarios. Furthermore, the current focus is on DDoS Miari attacks. Although these attacks are known to be widespread and significantly impact IoT infrastructure, other threats pose an equally significant danger. To mitigate these limitations and ensure a more accurate and complete representation of benign traffic data and potential threats, the next step in our research involves integrating data from existing datasets, e.g., the TON-IoT dataset [37]. This initiative will enrich our framework with diverse traffic protocols and consider a wider range of threats to improve analysis validity and reliability.

A further aspect to take into consideration is that the testbed's simplified environments may not fully mirror real IoT systems' complexities. Future work will take into account more dynamic and variable network conditions, such as variable latency and real-time traffic congestion, to help improve the realism of the simulations. Moreover, in the current study, we investigate a limited set of algorithms, i.e., RF, k-means, and CNN. However, we consider extending the investigation through a more in-depth analysis, which aims to explore the performance, computational resource, and memory requirements of additional ML models representative of the most popular tools used for intrusion detection in the IoT domain (e.g., Support Vector Machine (SVM), Isolation Forest (IF), Variational Autoencoder (VAE)). This initiative aims to more precisely identify an optimal algorithm that combines high performance and efficient resource consumption, thus outlining an ideal profile for an IoT context that inevitably has resource constraints.

Despite these potential limitations, we believe that DDOSHIELD-IoT provides a valuable platform for IDS evaluation and experimentation, offering insights into the capabilities and limitations of ML-based intrusion detection systems in IoT environments.

VI. CONCLUSION

This paper introduces DDOSHIELD-IoT, a simulation testbed for IDSs designed to assist security researchers. DDOSHIELD-IoT facilitates the generation of realistic traffic and the evaluation of ML models for the real-time detection of botnet DDoS attacks. Leveraging the foundation provided by DDoSim, DDOSHIELD-IoT employs the NS-3 network simulator and Docker container integration. This setup enables the hosting of diverse IoT binaries and the generation of real-world benign and malicious traffic. Such capabilities allow for the in-depth analysis of IDSs within a controlled environment. Our evaluation of DDOSHIELD-IoT highlights its effectiveness in real-time botnet DDoS attack detection using ML algorithms such as Random Forest, K-Means, and CNN with high accuracy. DDOSHIELD-IoT is publicly available at https://github.com/iobaidat/DDoShield-IoT, supporting the security community in the development of robust IDS solutions for IoT networks.

In future work, we will focus on Green *Artificial Intelligence* (AI) and *Federated Learning* (FL). Green AI initiatives to develop energy-efficient AI systems, potentially reducing energy consumption in IoT devices used for network monitoring and analysis [38]. FL presents a promising solution to address security, privacy, and sustainability concerns in IoT Intrusion Detection. Therefore, our upcoming objective is to enhance DDOSHIELD-IoT to emulate a FL-based *Network Intrusion Detection System* (NIDS) in line with Green AI principles, ensuring high accuracy based on the ML model identified in our study.

ACKNOWLEDGMENT

This work has been partially supported by MIUR for research activities - PON Research and Innovation 2014-2020; Action IV.5 "Doctorates on green topics" - CDA Resolution No. 73 of 29.04.2022. UGOV: 000010—PON_-DOTT_GREEN_ITEE_37_CICLO_001_005, and by the GENIO project (CUP B69J23005770005) funded by MIMIT.

REFERENCES

[1] S. Nižetić, P. Šolić, D. L.-d.-I. Gonzalez-De, L. Patrono *et al.*, "Internet of things (iot): Opportunities, issues and challenges towards a smart and sustainable future," *Journal of cleaner production*, vol. 274, p. 122877, 2020.

[2] K. Kimani, V. Oduol, and K. Langat, "Cyber security challenges for iot-based smart grid networks," *International journal of critical infrastructure protection*, vol. 25, pp. 36–49, 2019.

[3] M. Roopak, G. Y. Tian, and J. Chambers, "An intrusion detection system against ddos attacks in iot networks," in *2020 10th annual computing and communication workshop and conference (CCWC)*. IEEE, 2020, pp. 0562–0567.

[4] I. Stellios, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, and J. Lopez, "A survey of iot-enabled cyberattacks: Assessing attack paths to critical infrastructures and services," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3453–3495, 2018.

[5] A. Khraisat and A. Alazab, "A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges," *Cybersecurity*, vol. 4, pp. 1–27, 2021.

[6] A. Heidari and M. A. Jabraeil Jamali, "Internet of things intrusion detection systems: a comprehensive review and future directions," *Cluster Computing*, vol. 26, no. 6, pp. 3753–3780, 2023.

[7] M. Conti, D. Donadel, and F. Turrin, "A survey on industrial control system testbeds and datasets for security research," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2248–2294, 2021.

[8] J. Bhayo, S. A. Shah, S. Hameed, A. Ahmed, J. Nasir, and D. Draheim, "Towards a machine learning-based framework for ddos attack detection in software-defined iot (sd-iot) networks," *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106432, 2023.

[9] M. Zolanvari, M. A. Teixeira, L. Gupta, K. M. Khan, and R. Jain, "Machine learning-based network vulnerability analysis of industrial internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6822–6834, 2019.

[10] E. Anthi, L. Williams, M. Słowińska, G. Theodorakopoulos, and P. Burnap, "A supervised intrusion detection system for smart home iot devices," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 9042–9053, 2019.

[11] L. Nie, Z. Ning, X. Wang, X. Hu, J. Cheng, and Y. Li, "Data-driven intrusion detection for intelligent internet of vehicles: A deep convolutional neural network-based method," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2219–2230, 2020.

[12] S. Lee, S. Lee, H. Yoo, S. Kwon, and T. Shon, "Design and implementation of cybersecurity testbed for industrial iot systems," *The Journal of Supercomputing*, vol. 74, pp. 4506–4520, 2018.

[13] V. Zieglmeier, S. Kacianka, T. Hutzelmann, and A. Pretschner, "A real-time remote ids testbed for connected vehicles," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, 2019, pp. 1898–1905.

[14] K. Albulayhi, Q. Abu Al-Haija, S. A. Alsuhibany, A. A. Jillepalli, M. Ashrafuzzaman, and F. T. Sheldon, "Iot intrusion detection using machine learning with a novel high performing feature selection method," *Applied Sciences*, vol. 12, no. 10, p. 5015, 2022.

[15] A. Kumar and T. J. Lim, "Edima: Early detection of iot malware network activity using machine learning techniques," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*. IEEE, 2019, pp. 289–294.

[16] G. Zachos, G. Mantas, I. Essop, K. Porfyrakis, J. M. C. Bastos, and J. Rodriguez, "An iot/iomt security testbed for anomaly-based intrusion detection systems," in *2023 IFIP Networking Conference (IFIP Networking)*. IEEE, 2023, pp. 1–6.

[17] M. Eskandari, Z. H. Janjua, M. Vecchio, and F. Antonelli, "Passban ids: An intelligent anomaly-based intrusion detection system for iot edge devices," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 6882–6897, 2020.

[18] I. Obaidat, B. Kahn, F. Tavakoli, and M. Sridhar, "Creating a large-scale memory error iot botnet using ns3dockeremulator," in *2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2023, pp. 470–479.

[19] A. Boukhamla and J. C. Gaviro, "Cicids2017 dataset: performance improvements and validation as a robust intrusion detection system testbed," *International Journal of Information and Computer Security*, vol. 16, no. 1-2, pp. 20–32, 2021.

[20] M. Wu, J. Song, L. W. L. Lin, N. Aurelle, Y. Liu, B. Ding, Z. Song, and Y. B. Moon, "Establishment of intrusion detection testbed for cybermanufacturing systems," *Procedia Manufacturing*, vol. 26, pp. 1053–1064, 2018.

[21] S. Siboni, V. Sachidananda, Y. Meidan, M. Bohadana, Y. Mathov, S. Bhairav, A. Shabtai, and Y. Elovici, "Security testbed for internet-of-things devices," *IEEE transactions on reliability*, vol. 68, no. 1, pp. 23–44, 2018.

[22] R. Arthi and S. Krishnaveni, "Design and development of iot testbed with ddos attack for cyber security research," in *2021 3rd International Conference on Signal Processing and Communication (ICPSC)*. IEEE, 2021, pp. 586–590.

[23] L. E. S. Jaramillo, "Malware detection and mitigation techniques: Lessons learned from mirai ddos attack," *Journal of Information Systems Engineering & Management*, vol. 3, no. 3, p. 19, 2018.

[24] L. H. Newman. (2016) What we know about friday's massive east coast internet outage. [Online]. Available: https://www.wired.com/author/lily-hay-newman/

[25] J. Brownlee, *Data preparation for machine learning: data cleaning, feature selection, and data transforms in Python*. Machine Learning Mastery, 2020.

[26] N. Dong, L. Zhao, C.-H. Wu, and J.-F. Chang, "Inception v3 based cervical cell classification combined with artificially extracted features," *Applied Soft Computing*, vol. 93, p. 106311, 2020.

[27] T. Ma, F. Wang, J. Cheng, Y. Yu, and X. Chen, "A hybrid spectral clustering and deep neural network ensemble algorithm for intrusion detection in sensor networks," *Sensors*, vol. 16, no. 10, p. 1701, 2016.

[28] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, 2016, pp. 21–26.

[29] M. Kuhn, K. Johnson *et al.*, *Applied predictive modeling*. Springer, 2013, vol. 26.

[30] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems*, vol. 33, no. 12, pp. 6999–7019, 2021.

[31] K. P. Sinaga and M.-S. Yang, "Unsupervised k-means clustering algorithm," *IEEE access*, vol. 8, pp. 80 716–80 727, 2020.

[32] A. Parmar, R. Katariya, and V. Patel, "A review on random forest: An ensemble classifier," in *International conference on intelligent data communication technologies and internet of things (ICICI) 2018*. Springer, 2019, pp. 758–763.

[33] M. S. Murshed, C. Murphy, D. Hou, N. Khan, G. Ananthanarayanan, and F. Hussain, "Machine learning at the network edge: A survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 8, pp. 1–37, 2021.

[34] M. S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, and A. P. Sheth, "Machine learning for internet of things data analysis: A survey," *Digital Communications and Networks*, vol. 4, no. 3, pp. 161–175, 2018.

[35] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[36] C. Zhu, V. C. Leung, L. Shu, and E. C.-H. Ngai, "Green internet of things for smart world," *IEEE access*, vol. 3, pp. 2151–2162, 2015.

[37] N. Moustafa, "Ton_iot datasets," 2019. [Online]. Available: https://dx.doi.org/10.21227/fesz-dm97

[38] M. A. Albreem, A. M. Sheikh, M. H. Alsharif, M. Jusoh, and M. N. M. Yasin, "Green internet of things (giot): applications, practices, awareness, and challenges," *IEEE Access*, vol. 9, pp. 38 833–38 858, 2021.