IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 24 August 2023, accepted 20 September 2023, date of publication 25 September 2023,
date of current version 31 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3319213

RESEARCH ARTICLE

TII-SSRC-23 Dataset: Typological Exploration of Diverse Traffic Patterns for Intrusion Detection

DANIA HERZALLA, WILLIAN TESSARO LUNARDI, (Member, IEEE), AND MARTIN ANDREONI, (Member, IEEE)
Technology Innovation Institute, Masdar City, Abu Dhabi, United Arab Emirates

Corresponding author: Willian Tessaro Lunardi (willian.lunardi@tii.ae)

ABSTRACT The effectiveness of network intrusion detection systems, predominantly based on machine learning, is highly influenced by the dataset they are trained on. Ensuring an accurate reflection of the multifaceted nature of benign and malicious traffic in these datasets is paramount for creating IDS models capable of recognizing and responding to a wide array of intrusion patterns. However, existing datasets often fall short, lacking the necessary diversity and alignment with the contemporary network environment, thereby limiting the effectiveness of intrusion detection. This paper introduces TII-SSRC-23, a novel and comprehensive dataset designed to overcome these challenges. Comprising a diverse range of traffic types and subtypes, our dataset is a robust and versatile tool for the research community. Additionally, we conduct a feature importance analysis, providing vital insights into critical features for intrusion detection tasks. Through extensive experimentation, we also establish firm baselines for supervised and unsupervised intrusion detection methodologies using our dataset, further contributing to the advancement and adaptability of IDS models in the rapidly changing landscape of network security. Our dataset is available at https://kaggle.com/datasets/daniaherzalla/tii-ssrc-23.

INDEX TERMS Network traffic dataset, intrusion detection, network security, anomaly detection, machine learning.

I. INTRODUCTION

As the digital world becomes increasingly interconnected, and the need for robust network security has become paramount. This increasing interconnectedness, driven by technologies ranging from mobile computing to the Internet of Things (IoT), brings with it an exponentially growing attack surface, making network security not merely an optional layer but a critical necessity. At the heart of this defense strategy lie Intrusion Detection System (IDS). These systems employ many techniques, from statistical anomaly detection to signature-based methods and, increasingly, Machine Learning (ML) approaches, to identify and mitigate anomalous or malicious activity within a network. When discussing the role of ML in IDS, it's crucial to highlight the concept of data diversity, illustrated by practices like data augmentation. Data augmentation is a common technique to introduce variability into the training data in training ML models, particularly Deep Learning (DL) methods. This technique can prevent models from overfitting specific patterns and instead promote the ability to generalize to unseen instances. Similarly, the value of data diversity extends to network traffic datasets used for training IDS models, as it can enrich the models' ability to identify a broader range of intrusion scenarios.

Despite the critical importance of data diversity, traditional network traffic datasets, which are frequently employed in shaping network security approaches, exhibit significant limitations, most notably a lack of variation within the category of malicious samples, as documented in Appendix B Table 5. The table demonstrates that for 17 out of the 18 reviewed datasets, less than 10 attacks were captured. We aimed to contribute to this lack of depth in the diversity of attacks by launching 24 unique attacks. The lack of diversity, particularly within the malicious class, limits the ability of IDS models trained on these datasets to generalize effectively

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

to new, unseen intrusions commonplace in today's complex networks. The IoT has added another layer of complexity to network traffic, with its unique data patterns and its inherent security challenges. Despite efforts to create IoT-specific datasets, many of these initiatives fail to capture the full spectrum of device interactions and the diverse range of potential intrusions that can occur in these settings. The heterogeneity of IoT networks, characterized by a vast array of interconnected devices with varying capabilities and vulnerabilities, amplifies the challenge of curating a representative dataset. Consequently, this presents an urgent call for creating more comprehensive and diverse datasets that better encapsulate the contemporary threats networked systems face.

In this paper, we propose TII-SSRC-23, a new dataset designed to address the challenges outlined earlier. The dataset totals 27.5 GB and is bifurcated into two main categories: benign and malicious, encompassing eight distinct traffic types. These types are divided into 32 traffic subtypes: six benign and 26 malicious. Both the raw network traffic data, stored as Packet Capture (PCAP) files, and the extracted features, presented in the form of Comma-Separated Values (CSV) files, are included in our dataset. Our methodology for dataset generation begins with defining the network topology, serving as the foundation for all subsequent interactions. This includes generating benign traffic miming typical network interactions across unique data types such as video, audio, text, and background traffic. Following this, we outline the generation of malicious traffic, replicating four types of network threats: Denial of Service (DoS) attacks, brute-force attacks, information gathering tactics, and botnet traffic, with a specific emphasis on the Mirai botnet. Feature extraction and importance are analyzed, followed by supervised and unsupervised experiments that establish firm baselines for future works. Our main contributions can be summarized as follows:

- We present the open-source TII-SSRC-23 dataset, a heterogeneous collection encompassing eight traffic types (audio, background, text, video, bruteforce, DoS, information gathering, botnet) and 32 subtypes across both benign and malicious categories.
- We conduct an exhaustive survey on 18 existing network traffic datasets, providing key insights to aid researchers in dataset selection for IDS research.
- We perform a comprehensive feature importance analysis within network traffic data, offering valuable insights on critical features for intrusion detection tasks, thereby facilitating IDS model optimization.
- Through extensive experimental evaluation, we establish firm baselines for supervised and unsupervised intrusion detection methodologies using our dataset, fostering the development of robust IDS systems optimized for diverse network traffic situations.

The remainder of this paper is structured as follows: Section II provides a comprehensive review and analysis of preceding work that centers around creating and publicly releasing network traffic datasets, tackling the limitations and challenges inherent to existing data sources. Section III provides an exhaustive description of our proposed network IDS dataset generation process, encompassing the testbed, the types, and the characteristics of both benign and malicious traffic. In Section IV, we examine statistical patterns and characteristics of the produced network traffic through the lens of feature importance analysis. This includes data pre-processing stages, feature extraction via CICFlowMeter [1], and feature importance computations to discern the most informative features. Section V is dedicated to evaluating both supervised and unsupervised methodologies to set solid baseline performances for intrusion detection using our dataset. Conclusively, Section VII wraps up the paper.

II. RELATED WORKS

This section delves into a comprehensive timeline of IDS datasets spanning the last quarter-century, from earlier published datasets in 1998 to more recent ones released in 2023. We review a range of datasets, including some of the more traditional testbed datasets featuring network-layer attacks, real-world network deployments, and IoT datasets. Table 1 presents a survey of the datasets, considering characteristics such as the year of the dataset's creation, number of traffic objects, dataset's published format, size of the raw traffic, number of features extracted from the dataset, traffic source, and deployed network topology. The number of traffic objects is either represented as a value with the bidirectional flows[1] label (bi. flows) or just as a value. The latter implies that no information was found regarding the type of traffic object of the dataset. The published format, which represents the form in which the data was published, is described either as raw, denoting that the network traffic provides packet-level information, or as statistics, providing information about the traffic objects. The traffic source falls into three categories: real, emulated, or synthetic. Real denotes that the data were captured in a real-world network deployment, emulated refers to the data being captured in a controlled network environment with traffic generated manually using scripts or similar means, and synthetic means a network traffic simulation tool was used to generate data. In both real and emulated traffic, real network traffic is generated using real devices. The difference between both is related to the way in which the real network traffic is generated; in emulated traffic, the type of traffic generated in the order and time in which it is generated is controlled, whereas in real traffic the data is produced by users in an uncontrolled real-world deployed network environment. Finally, for the testbed, we have defined small to indicate that the testbed contained fewer than 20 nodes, medium to indicate that the testbed contained 20 to 50 nodes, and large to indicate that a real-world network deployment or a testbed consisting of more than 50 nodes was used. In the

[1] Formal definitions of unidirectional and bidirectional network flows can be found in Appendix A.

TABLE 1. IDS datasets characteristics.

Dataset	Year	# Traffic Objects	Published Format	Size (GB)	Features	Traffic Source	Testbed
DARPA98 [2]	1998	–	Raw	4	–	Emulated	Small (military)
KDD99 [3]	1998	4.9M bi. flows	Statistics	–	41	Emulated	Small (military)
NSL-KDD [4]	1998	1M bi. flows	Statistics	–	41	Emulated	Small (military)
Kyoto 2006+ [5]	2006-09	93M bi. flows	Statistics	–	24	Real	Large (honeypots)
UNIBS [6]	2009	79k bi. flows	Raw, statistics	2.7	8	Real	Medium (university)
CTU-13 [7]	2011	81M bi. flows	Raw, statistics	77	14	Real	Large (university)
TUIDS [8]	2011-12	250k bi. flows	Raw, statistics	–	50, 24	Real	Large (university)
ISCX 2012 [9]	2012	2M bi. flows	Raw, statistics	84.1	14	Emulated	Small
UNSW-NB15 [10]	2015	2.5M bi. flows	Raw, statistics	99.1	49	Synthetic	Small
DDoS 2016 [11]	2016	2.1M bi. flows	Statistics	–	27	Synthetic	–
CICIDS 2017 [12]	2017	3.1M bi. flows	Raw, statistics	47.9	80	Emulated	Medium
CIC DoS [13]	2017	–	Raw	4.6	–	Emulated	Small
N-Baiot [14]	2018	7M	Statistics	–	115	Emulated	Small (IoT)
BoT-IoT [15]	2019	73M bi. flows	Raw, statistics	69.4	46	Emulated, synthetic	Small (IoT)
TON-IoT [16]	2019	22M bi. flows	Raw, statistics	65.1	44	Emulated, synthetic	Medium (IoT)
CIC IoT [17]	2022	30k bi. flows	Raw, statistics	60.3	48	Emulated	Medium (IoT)
LATAM-DDoS-IoT [18]	2022	49M bi. flows	Raw, statistics	279.8	20	Real, emulated	Large (IoT)
Edge-IIoTset [19]	2022	20M bi. flows	Raw, statistics	69.3	61	Emulated	Medium (IIoT)
TII-SSRC-23 (ours)	2022-23	8.6M bi. flows	Raw, statistics	27.5	75	Emulated	Small

case that we could not find specific information for a dataset or is irrelevant considering the data available, it is indicated by a dashed mark.

The DARPA98 dataset [2] established a performance benchmark for intrusion detection systems with a military network testbed showcasing diverse traffic types like DoS, probing, and privilege escalation attacks. This dataset inspired the development of the KDD99 dataset [3], which processed the raw traffic portion of the DARPA98 dataset comprising of benign and malicious traffic. Despite its merits, KDD99 had a significant problem of redundant records [4], leading to the inception of the NSL-KDD dataset [4]. NSL-KDD, a polished version of KDD99, underwent preprocessing to eliminate redundancy, offering a more realistic evaluation context for intrusion detection systems and anomaly detection algorithms. However, these datasets share a key limitation – their outdatedness hinders their utility for modern network traffic analysis [20]. The Kyoto 2006+ dataset [5], which encapsulates real-world network traffic data harvested from Kyoto University between 2006 and 2009 using honeypots, has its limitations. It lacks manual labeling and introduces anonymization, and its network traffic perspective is constrained to honeypot-targeted attacks. While the dataset incorporates ten additional attributes compared to the aforementioned datasets that are useful for IDS investigation, the benign traffic simulation is limited to Domain Name System (DNS) and mail traffic data, excluding a more extensive range of real-world benign traffic.

The ISCX 2012 dataset [9] used an innovative approach involving α and β profiles to mimic benign user activities and malicious scenarios. The benign user behavior included traffic from the protocols: Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), Secure Shell Protocol (SSH), Internet Message Access Protocol (IMAP), Post Office Protocol (POP3), and File Transfer Protocol (FTP). This dataset includes raw packet-level data in PCAP files, featuring approximately 2.4 million bidirectional flows. Echoing this methodology, the CICIDS2017 dataset [12] generated a realistic background traffic scenario using the B-Profile system. This system models the behavior of 25 users based on HTTP, Hypertext Transfer Protocol Secure (HTTPS), FTP, SSH, and email protocols. It comprises six attack profiles, specifically bruteforce, heartbleed botnet, DoS, Distributed Denial of Service (DDoS), web, and infiltration attacks. Developed in 2015, the UNSW-NB15 dataset [10] comprises benign and malicious network traffic data generated using a network traffic simulation tool over a week in a controlled setting. The dataset includes nine attack classes: backdoors, DoS, exploits, fuzzers, and worms. Presented in packet-based format (PCAP) and bidirectional flow-based format, it features 49 attributes and predefined train-test splits. The dataset contains around 2.5 million bidirectional flows with an estimated 2.8% malicious traffic. The UNIBS dataset [6] consists of traffic collected on the edge router of a campus network using 20 workstations. The traffic collected provides valuable network traffic information related to the campus network's communication patterns and behavior. However, the dataset does not contain malicious traffic traces. The CTU-13 (Capture The Flag) dataset [7] contains real botnet traffic mixed with benign traffic captured in a university network. The malicious traffic includes 13 scenarios of botnet samples in which each scenario included botnet, benign, C&C, and background flows. The dataset is labeled to indicate the type of malware attack. It is available in PCAP and bidirectional flow-based format. The TUIDS dataset [8] encompasses benign user behavior and various malicious traffic types including botnet, DoS/DDoS, probing, coordinated port scan, and privilege escalation. The dataset was generated using approximately 250 clients, captured in raw packet-level and bidirectional flow formats. It is labeled and contains around 250k flows. As the dataset is not publically available, we could not determine the size of the raw traffic. Shifting the focus to DoS- and DDoS-based datasets, the DDoS 2016 dataset [11] contains benign traffic instances and focuses on DDoS attacks such as User Datagram Protocol (UDP) flood, smurf, HTTP flood, and SQL Injection Dos (SIDDoS). However, the traffic was generated using a network traffic simulator. The CIC DoS dataset [13] focuses on eight different application layer DoS attacks, particularly HTTP DoS. To create benign traffic that mimics normal user behavior, traffic from the ISCX 2012 dataset was used. The dataset is provided in raw capture format, making it useful for studying and evaluating intrusion detection methods in the context of application layer HTTP DoS attacks.

As for IoT-based datasets, the BoT-IoT dataset [15] offers a mix of benign and botnet traffic, simulating a realistic network environment. It comprises synthetically created benign traffic as well as diverse attack types such as DDoS, DoS, Operating System (OS) and service scan, keylogging, and data exfiltration attacks, with DDoS and DoS attacks further classified by protocol. The dataset incorporates protocols like Transmission Control Protocol (TCP), UDP, Address Resolution Protocol (ARP), Internet Control Message Protocol (ICMP), Internet Group Management Protocol (IGMP), and Reverse Address Resolution Protocol (RARP). The dataset features around 73 million bidirectional flows. The LATAM-DDoS-IoT dataset [18] is designed with a primary focus on DoS and DDoS attacks, implemented in a testbed of physical and virtual IoT components. Benign traffic from a production network was collected. The dataset includes two versions: LATAM-DoS-IoT and LATAM-DDoS-IoT, with 30 and 49 million bidirectional flows, respectively. The CIC IoT 2022 dataset [17] was developed for the profiling, behavioral analysis, and vulnerability testing of IoT devices using various protocols. It collects data from experiments covering power-on, idle, interactions, scenarios, active network communications, and attack traffic: flood and Real Time Streaming Protocol (RTSP) bruteforce. The collection process targeted IoT devices linked to an unmanaged switch, simulating a wireless IoT environment. The Edge-IIoTset dataset [19] caters to IoT and Industrial Internet of Things (IIoT) applications. The dataset is a multi-layered testbed, utilizing more than 10 different IoT devices, and encompasses 14 attacks related to IoT and IIoT connectivity protocols. These attacks are categorized into five threats, including DoS and DDoS, information gathering, injection, man-in-the-middle, and malware attacks. The dataset contains around 20 million bidirectional flows, with about 11.2 million benign and 9.7 million malicious, with 61 extracted traffic features. The TON-IoT dataset [16] integrates IoT and IIoT systems and devices across edge, fog, and cloud layers within an orchestrated testbed architecture. The data encapsulate both synthetically created benign traffic and nine attack scenarios, shared as raw and processed traffic data in PCAP and CSV formats, along with operating system logs. The dataset comprises approximately 22.3 million bidirectional flows captured in 44 features. The benign traffic represents around 3.6% of the flows in the dataset, leaving about 96.4% as malicious flows. Lastly, the N-BaIoT dataset [14], captured in an IoT lab environment, records benign and botnet events. The dataset includes network traffic data from nine IoT devices and encompasses 10 attack types originating from the BASHLITE and Mirai botnets. Featuring 23 distinct features, the dataset is shared in CSV format. The dataset comprises over 7 million flows.

Table 5 lists the attacks executed in all the aforementioned IDS datasets. Although multiple datasets exist, such as UNSW-NB15 and CICIDS 2017, encompass many attack categories, our dataset concentrates on a wide breadth of each attack. Specifically, we investigate a variety of attacks within each of our four categories: DoS, bruteforce, information gathering, and botnet. This investigation results in a total of 26 unique attacks launched.

III. TII-SSRC-23: DATASET GENERATION METHODOLOGY

In this section, we detail our methodology for creating the proposed 27.5 GB dataset in PCAP format. The traffic is bifurcated into two primary categories (benign and malicious), spanning eight traffic types (audio, background, text, video, bruteforce, DoS, information gathering, Mirai botnet), including 32 subtypes (six benign and 26 malicious). Table 2 identifies the traffic types and subtypes, with each subtype quantified by the number of combinations[2] and bidirectional flows. Moreover, the "combinations" column denotes the traffic variations within a traffic subtype, approximated by the number of traffic permutations launched informed by the subtype's parameters, as listed in Appendix Section B. The necessity for diversifying traffic patterns to enhance the resilience of IDS is examined in Section III-A. Our methodology begins with the specification of the network topology, outlined in Section III-B, which forms the foundation for all subsequent interactions. The generation of benign traffic, emulating typical network interactions across the following unique data types: video, audio, text, and background traffic, is illustrated in Section III-C. Finally, Section III-D describes the generation of malicious traffic, replicating four types of network threats.

A. TRAFFIC DIVERSIFICATION FOR IMPROVED IDS ROBUSTNESS

Despite the impressive performance of various IDS datasets evaluated through ML/DL methodologies within their corresponding test environments, a significant performance decline is observed when these models are implemented in real-world contexts [20]. This performance degradation often results in expensive misclassifications due to high false positive or false negative rates, thereby underlining a predominant challenge encountered by ML-driven IDSs.

[2] The number of combinations can exceed the number of bidirectional flows; this strictly depends on the protocol and how they are terminated.

TABLE 2. Distribution of bidirectional network traffic flows in the dataset, classified by type and subtype. The column "Bi. Flows" represents the number of samples for each traffic subtype.

Cat.	Type	Subtype	Combinations	Bi. Flows
Benign	Audio	Audio	1	190
	Background	Background	1	32
	Text	Text	1	209
	Video	HTTP	180	376
		RTP	180	349
		UDP	180	145
Malicious	Bruteforce	DNS	2	22179
		FTP	1	3485
		HTTP	2	628
		SSH	1	3967
		Telnet	1	4913
	DoS	ACK	24	936307
		CWR	24	872523
		ECN	24	871150
		FIN	24	725600
		HTTP	27	82351
		ICMP	16	9
		MAC	1	30
		PSH	24	909507
		RST	24	1072504
		SYN	24	856764
		UDP	24	257994
		URG	24	906190
	Information Gathering	Information Gathering	102	1038363
	Mirai	DDoS ACK	3	3779
		DDoS DNS	1	55196
		DDoS GREETH	6	43
		DDoS GREIP	6	49
		DDoS HTTP	8	8923
		DDoS SYN	12	14210
		DDoS UDP	6	71
		Scan and Bruteforce	1	8731

An effective mitigation strategy involves utilizing network traffic datasets with diversified characteristics during training. This diversification allows the models to generalize better and accurately classify network traffic in real-world deployments. Although numerous existing datasets underscore the incorporation of an extensive variety of benign and malicious traffic, the emphasis on including diverse traffic patterns within each traffic category is noticeably lacking. In contrast, our proposed IDS dataset adopts a unique approach by stressing the generation of diversified traffic patterns within each traffic category. This is achieved through carefully manipulating data traffic parameters during the data generation stage, as described in the subsequent sections. By integrating this degree of diversity, our dataset is designed to enhance the robustness and effectiveness of ML-based IDSs, particularly when facing an array of complex and evolving network traffic situations.

B. NETWORK CONFIGURATION OVERVIEW

Our data recording setup captured benign and malicious traffic, deploying a testbed configuration composed of five nodes. These nodes encompass two laptop systems running Ubuntu 20.04 and three embedded devices, each offering processing capabilities equivalent to a Compute Module 4 device.[3] Two of the embedded devices are interconnected to each laptop via Ethernet connections. At the same time, the third embedded device operated as a mobile unit, allowing placement in various locations, thus facilitating the simulation of diverse network interference scenarios. During traffic recording, the mobile embedded device is strategically relocated across three distinct locations to generate variations in network interference. The labels "low," "mid," and "high" interference, which are relative terms, denote the distinct degrees of interference experienced at each respective location, as determined by the corresponding throughput values, i.e., approximately 154 Megabits per second (Mbps) for "low", 69.7 Mbps for "mid", and 38.4 Mbps for "high" interference scenarios. Specifically, at the first location the mobile device is placed half a meter away from the testbed, leading to the lowest level of interference. At the second location, the mobile device is stationed six meters horizontally away from the testbed, separated by two rooms, resulting in the highest level of interference. In contrast, the third location sees the mobile device placed six meters below the testbed, precisely one floor apart with a glass wall and concrete flooring, creating mid-level interference. During all traffic capture scenarios, the tcpdump tool[4] was set to capture the traffic on the mobile embedded device. The embedded devices operate within a decentralized system where peer-to-peer communication occurs via a wireless medium. The traffic flow path is managed via the Better Approach to Mobile Ad-hoc Networking (BATMAN) [21] protocol chain, maintaining a static bi-directional path. This setup ensures that the communication passes through all nodes within the BATMAN chain before reaching the destination node.

In the Mirai malware attack scenario, the communication between the Command-and-Control (CnC) server and the bots does not follow an end-to-end path. Consequently, to comprehensively capture all CnC and botnet traffic we recognized the need to construct a centralized testbed. This modified testbed included five nodes, with a Raspberry Pi 4 set as the Access Point, two Ubuntu 20.04 laptop systems as the victim and the botmaster hosting the CnC server, and the ScanListen server, as well as two bots deployed on Compute Module 4 (CM4) boards. All traffic was recorded on the Access Point using the tcpdump tool to capture bidirectional communication between the botmaster, the bots, and the victim.

C. BENIGN TRAFFIC

Our data collection, within the context of benign traffic, comprises four distinct types: audio, video, text, and background. Video traffic comprises the majority of benign flows,

[3] Raspberry Pi Compute Module 4. https://datasheets.raspberrypi.com/cm4/cm4-datasheet.pdf

[4] Tcpdump: Unix-based network packet analyzer http://www.tcpdump.org/

accounting for more than 65% as deduced from Table 2. Audio and text traffic each comprises around 15%, with background traffic making up around 3%.

1) AUDIO AND TEXT TRAFFIC

The Mumble[5] voice-over Internet Protocol (IP) application was utilized to create audio and text traffic independently. The interaction between the client and the server was enabled using the Pymumble Python module, with a Python script devised to transmit audio and text messages using a script with over 100 varied-length strings from screenplays and literary works; the strings contained a variation of alphanumeric and special characters. The network environment incorporated one server and three clients. The server was set on an embedded device, and the two laptop machines operated as clients, transmitting messages to the server. The clients dispatched audio/text messages with a 5% probability of disconnection from the server. Upon disconnection, the client system was programmed to automatically re-establish the connection after a brief intermission. The audio and text traffic were each captured over a period of one hour.

2) BACKGROUND TRAFFIC

The background traffic was recorded for a period of one hour. This strategy was twofold: not only did it contribute to the dataset by gathering background traffic, but it also provided a reference framework to aid in manually identifying specific background data types requiring filtration from the attack PCAP files. The background traffic contains broadcast packets, DNS messages, ICMP router solicitation messages, and ARP packets; the protocols spanned the following: ATA over Ethernet (AOE)), multicast Domain Name Resolution (MDNS), Internet Control Message Protocol for IPv6 (ICMPv6), and ARP.

3) VIDEO TRAFFIC

The Video LAN Client (VLC) application was employed to generate video traffic, leveraging its accompanying Python module for automated video streaming. A custom Python script was created to introduce heterogeneity in the video traffic by modulating ten distinct video streaming parameters: pixel resolution, video codec, audio codec, video bitrate, audio bitrate, video scale, frames per second, multiplexer type, sample rate, and the underlying protocol. The VLC streaming server was instantiated on the laptop. This server was responsible for streaming a playlist of seven unique videos. The streaming session was allotted one hour, where protocols such as UDP, Real-Time Transport Protocol (RTP)/Transport Stream (TS), and HTTP were utilized for transmission. This procedure led to the creation of a PCAP file for each of the utilized communication protocols. Comprehensive details regarding the modulated video traffic parameters are available in Appendix B Table 6.

D. MALICIOUS TRAFFIC

In the context of malicious traffic, our compiled dataset embodies four different attack types. These comprise DoS, Bruteforce, Information Gathering, and Botnet. The DoS attacks represent the majority, accounting for approximately 86% of the malicious traffic flows, followed by Information Gathering accounting for 12%. Mirai Botnet and bruteforce each constitute 1% of the malicious traffic. After the data capture, a filtering process was applied to the attack PCAP files during preprocessing, purging them of non-malicious data, as expanded upon in Section IV-A.

1) DOS

DoS attacks, regarded as one of the most pervasive and frequently exploited types of network traffic intrusions, have witnessed a surge in both frequency and intensity in recent years. The year 2015 was a notable milestone in the history of DoS attacks, setting unprecedented records for data flood transfer rates, a trend that intensified in the following year [22]. These attacks, infamous for their disruptive effects, can rapidly deplete their targets' computational resources and bandwidth within minutes, effectively denying access to legitimate users. Reflecting the significant relevance of these attacks and in line with this trend, more than 85% of our dataset constitutes DoS attacks. Our investigation covers 12 unique flood attacks, each exploiting distinct vulnerabilities to inundate target devices. These flood attacks span HTTP, ICMP, Media Access Control (MAC), TCP (Acknowledgement (ACK), Congestion Window Reduced (CWR), Explicit Congestion Notification (ECN), Finish (FIN), Push (PSH), Reset (RST), Synchronize (SYN), Urgent (URG)), and UDP. To incorporate variability and diversify the traffic, we meticulously modulated multiple parameters during the deployment of these attacks. Parameters such as speed of packet transmission and payload size can be key indicators of a DoS attack. Reference [11] given their significance in the identification of DoS activities, we dedicated special attention to manipulating them in order to capture the various ways these parameters are exploited by attackers. Within the ICMP, TCP, and UDP floods we adjusted the speed of packet transmission to three distinct modes specified in Hping3: "fast", "faster", and "flood", ranging from 10 packets per second (pps) to over 1000 pps, capturing a range of stealthy to aggressive flood attacks. Additionally, we varied the payload size to range from small inconspicuous payloads to larger payload flooding tactics to capture diverse DoS attack strategies.

The TCP flood attack capitalizes on the intrinsic features and behavior of the TCP protocol, exploiting the interactions using various flags present within the TCP packets. As listed above, we launched eight distinct types of TCP flood attacks. Within each, we varied six attack-related parameters: packet transmission speed, payload size, randomized source ports, TCP checksum validity, TCP window size, and TCP data offset. This yielded 192 unique TCP flood traffic combinations captured over 18.7 minutes. The UDP flood

[5] Mumble: open-source voice chat application. https://www.mumble.info/

attack operates by transmitting a large volume of UDP packets. We modulated four parameters: packet transmission speed, payload size, randomized source ports, and UDP checksum validity, producing eight UDP traffic combinations captured over a period of three minutes. In the case of the ICMP flood, we varied the payload size resulting in four unique combinations of traffic captured over a period of two minutes. The HTTP flood attack is a type of volumetric application layer attack that aims to inundate the target with HTTP requests. We modulated three parameters for this attack: request method (GET, POST, Random), number of concurrent workers, and number of concurrent sockets. This configuration resulted in 27 unique traffic streams spanning a period of 11.3 minutes. The MAC flood was launched for 30 minutes, with no parameters adjusted, as the macof tool does not provide any traffic options to vary. In Appendix Table 7, we provide further details of the modulated parameters for each flood attack, offering deeper insights into the experimental setup and configuration for our IDS dataset.

2) BRUTEFORCE

Despite their age and lack of sophistication, bruteforce attacks retain startling prevalence and efficacy in the contemporary digital landscape. This attack involves systematically attempting all possible combinations of credentials from a list of keys to discover a successful pair. The Patator tool[6] was used to execute bruteforce attacks on five services: DNS (forward and reverse lookup), FTP, HTTP, SSH, and Telnet. For launching the bruteforce attacks on the FTP, HTTP, SSH, and Telnet services, we used a list of around 400k usernames and two million leaked passwords.[7] The Filezilla Client application was set on the victim to perform the FTP bruteforce attack. The HTTP bruteforce attack was executed against a phpMyAdmin server hosted on the victim's machine, using GET and POST request methods. To carry out the forward DNS lookup, we tested around 12k domain names against the server domain. The reverse DNS lookup involved querying a range of IP addresses to identify the victim's hostname.

3) INFORMATION GATHERING

An information-gathering attack constitutes a critical initial step for attackers preparing for future exploits on their target system, proving particularly beneficial for malware attacks. Such an attack aims to acquire, among other things, information on a network's architecture, OS, and active security defense mechanisms. Information-gathering attacks manifest in several forms, of which we implemented six types, specifically: port scan (TCP and UDP), OS detection, version detection, script scan, and ping scan utilizing the Hping3 and Nmap[8] tools. We employed various IDS evasion strategies to circumvent detection to render the scans more covert.

A port scan involves scanning the ports of the victim to ascertain their status. The execution of a successful port scan provides the attacker with an entry point to penetrate the network and extract the targeted information. Hping3 was utilized to perform a scan on all ports using six TCP flags. Additionally, Nmap was used to perform a UDP scan and seven types of TCP port scans with multiple parameters varied for each. The TCP port scans were of the following types: Connect, SYN ACK, FIN, Window, Maimon, XMAS, and NULL. As for a ping scan, it operates to discern the presence of hosts in a network by using their IP addresses. Nmap was deployed to launch seven ping scans, namely: ICMP echo, ICMP timestamp request, ICMP netmask request, TCP SYN, TCP ACK, UDP, and Stream Control Transmission Protocol (SCTP) Initialization (INIT) scans. Finally, OS detection, version detection, script scanning, and traceroute techniques were performed. This was facilitated using the pre-configured ''Aggressive Scan'' Nmap option, which activates multiple advanced scans to probe the target machine comprehensively. All of the information gathering tactics yielded 102 unique combinations of traffic, elaborated upon in Appendix table 7.

4) BOTNET MALWARE

In the field of cybersecurity, malware–a form of software-based attack–poses a significant threat by compromising system confidentiality. This breach can lead to sensitive data theft, disruption in system operations, or render the system entirely inoperative. Among various types of cyberattacks against embedded systems, botnet malware is one of the most prevalent [23]. The Mirai botnet is a notable example of this type of malware [23]. Designed specifically to infiltrate devices running a Linux system, Mirai aims to transform these systems into botnets that can launch substantial network-level and HTTP flood attacks on servers. Mirai executes this by exploiting the default username and password combinations configured during the initialization of IoT devices. The common expectation is that users will replace these default credentials. However, this often does not occur in practice, leading to devices remaining vulnerable to malicious intrusion. In such cases, hackers leverage scanning and bruteforce attacks to identify accessible devices to gain control over the device by injecting the Mirai malware. The Mirai attack follows the following sequence of events: (*Scanning Stage*) The existing bots initiate a scan to identify potential new devices to infect. As the bots were deployed on two CM4 devices with limited processing power, the scanning process was significantly time-consuming. To expedite the bruteforce stage, we manually configured the target IP; (*Bruteforce Stage*) The bots then attempt to brute open Telnet

[6] Patator: multi-purpose bruteforcer https://www.kali.org/tools/patator/

[7] The list of credentials used were obtained from a bruteforce database. https://github.com/duyet/bruteforce-database/tree/master

[8] Nmap: open-source utility for network discovery and security auditing. https://nmap.org/

ports on discovered devices utilizing a set of commonly used IoT device credentials. Upon successful bruteforce attempts, the bots report the pertinent device details and the successful credentials to the ScanListen server; (*Loader Stage*) The CnC server monitors the status of the ScanListen server and instructs the loader to inject a malicious binary onto the discovered device upon successful authentication. The Mirai malware was manually loaded onto the CM4 bots as they were found to be immune to Mirai infection; (*Attack Stage*) The CnC server then dispatches attack commands to the bots to initiate an attack on a specific victim IP.

We initiated eight vectors of the Mirai attack, specifically: ACK, DNS, HTTP, GREETH, GREIP, SYN (SYN URG, SYN PUSH, SYN RST, SYN FIN, SYN-ACK), UDP, and UDP plain flood attacks. The UDP Plain flood attack is a simplified version of the UDP flood, offering limited options but enabling a higher packet transmission rate. The GREETH and GREIP attacks inundate the target with malicious Generic Routing Encapsulation (GRE) encapsulated Ethernet and IP packets, respectively. The GREETH assault includes Transparent Ethernet Bridging over GRE-encapsulated packets in its payload, whereas the GREIP attack encompasses solely IP packets. Despite similar operational patterns, the GREETH attack incorporates an additional L2 frame. We altered various attack parameters in initiating the Mirai DDoS assaults, some of which include the payload size, randomized source and destination ports, and type of service, as elaborated upon in Appendix Table 8. The resultant Mirai DDoS attack data comprise two primary traffic types: CnC traffic, capturing the interaction between the botmaster and the bots, as well as bot traffic, which represents the DDoS attack activities. We also share the scanning and bruteforce traffic between the bots and the target device.

IV. NETWORK TRAFFIC FEATURE EXTRACTION AND IMPORTANCE EVALUATION

This section is dedicated to exploring the procedures of feature extraction and importance evaluation in network traffic data. Our main interest lies in revealing inherent statistical tendencies and subtleties encapsulated in the network traffic data that have been generated. An overview of the data preprocessing stages, including the filtering of PCAP files, is given in Section IV-A. We employ the CICFlowMeter tool for feature extraction and elucidate this process in Section IV-B. In Section IV-C, we delve into feature importance analysis, providing an in-depth study of the most impactful features related to various types of network traffic.

A. DATA FILTERING AND PREPROCESSING

Following data capture, Wireshark was used to filter the obtained files, stored in the PCAP format, based on the type of traffic each contained. Files containing malicious data underwent manual filtering to eliminate background traffic, which helped prevent contamination of the malicious files with benign data. The background traffic PCAP helped determine what types of benign data packets the malicious traffic files needed to be filtered from. We noticed the rare presence of packets with random protocols in the files associated with DoS attacks. As these are presumably part of the executed attack, they were not filtered out.

B. FEATURE EXTRACTION

While the primary objective of this study is not to contribute to the field of feature engineering, it is essential to describe the process we employed to extract valuable insights from our network traffic data. We utilized CICFlowMeter, a well-acknowledged tool frequently employed in intrusion detection literature. CICFlowMeter establishes a robust framework for extracting crucial features from traffic sessions. These sessions are defined based on bidirectional flows, a strategy consistent with the predominant network traffic object used for classification, compared to packets and unidirectional flows. Bidirectional flows offer a comprehensive network traffic perspective, facilitating precise and detailed examination. CICFlowMeter enables us to extract 75 distinct features from each bidirectional flow. The tool processes raw network traffic data, maps the packets to their respective bidirectional flows, and then computes essential statistical features.[9] The processed data, represented in the form of these computed features, are provided in a structured CSV file format. This format streamlines the subsequent stages of network traffic data analysis and interpretation. The CSV files were labeled, incorporating three levels of classification such as "Label" (Benign or Malicious), "Traffic Type" (Audio, Background, Text, Video, Bruteforce, DoS, Information Gathering, Mirai), and "Traffic Subtype" as listed in Table 2.

To further understand the distribution and structure of our high-dimensional data, we employ t-distributed Stochastic Neighbor Embedding (t-SNE) for visualization. Figure 1 presents the t-SNE plot of our data, providing a clear visual summary of how our data points relate. From the plot, one can also discern the rich diversity inherent in the TII-SSRC-23 dataset. While distinct clusters corresponding to different traffic types are evident, the mingling of samples, especially within the malicious categories, underscores the multifaceted nature of intrusion patterns captured in our dataset. This intermingling, far from being a drawback, actually highlights the dataset's comprehensive coverage of a vast spectrum of attack vectors and behaviors.

C. FEATURE IMPORTANCE ANALYSIS

Before delving into the experimental phase of this study, it is critical to conduct a comprehensive analysis of feature importance. This analysis not only allows us to ascertain the relative significance of each feature and comprehend its bearing on the classification task but also provides insights for future work. Given the high dimensionality of our dataset,

[9] For more details regarding the extracted features, please refer to the CICFlowMeter Github repository: https://github.com/ahlashkari/CICFlowMeter/blob/master/ReadMe.txt

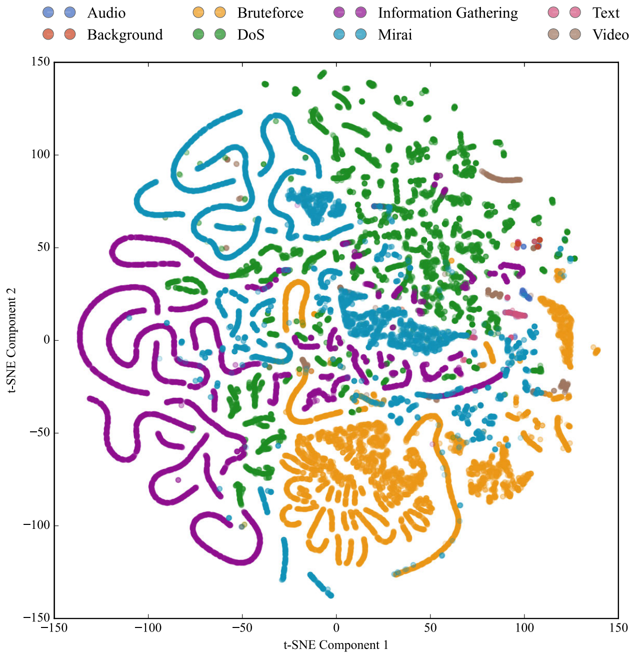
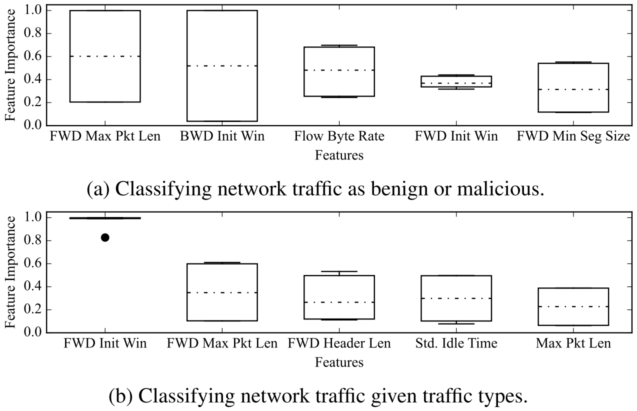


FIGURE 1. Clusters in network traffic data visualized using t-SNE.

pinpointing the features that contribute most profoundly to our classification models' performance is vital. Additionally, this analysis is instrumental for future research that utilizes our shared dataset, as it provides valuable insights into model development within intrusion detection. This foundational understanding of feature importance could be leveraged to enhance the effectiveness of future intrusion detection models and strategies.

We employed Permutation Feature Importance (PFI) to compute the feature importance. PFI works by randomly shuffling the values of one feature at a time and then evaluating the resultant effect on the model's performance. A marked decrease in the model's performance implies the shuffled feature's importance for the predictive task in question. However, evaluating feature importance should not entirely depend on a singular execution of PFI. It is advisable to perform multiple runs per method and utilize various classifiers when assessing feature importance. This is because a feature's importance can fluctuate depending on the model's architecture and the specific run of the algorithm. We promote a more comprehensive understanding of feature importance by employing multiple methods and runs, providing a more robust foundation for our analysis. We employed three classifiers to calculate feature importance: the Random Forest (RF) classifier, the eXtreme Gradient Boosting (XGBoost) classifier, and the Extra Trees (ET) classifier. These classifiers were selected by their efficiency and potential for parallelization, which permitted the experiment to be carried out within a feasible timeframe. It's also worth noting that, for each classifier, we conducted three separate runs of PFI, thereby enhancing the reliability of our feature importance estimations.



(a) Classifying network traffic as benign or malicious.

(b) Classifying network traffic given traffic types.

FIGURE 2. Ranking of the five most critical features in network traffic classification. Plot (a) illustrates the five attributes distinguishing benign from malicious traffic. Plot (b) depicts the five principal features employed in segregating network traffic into various unique categories: audio, video, text, DoS, Mirai, and bruteforce attacks.

Two distinct feature importance experiments were conducted: (1) a binary classification experiment aimed at distinguishing benign from malicious traffic and (2) a multiclass classification experiment intended to identify specific types of network traffic. Boxplots of the feature importances for each scenario are presented in Figure 2. Plot (a) displays the top five features in distinguishing benign traffic from malicious ones. In contrast, plot (b) outlines the five most influential features in segregating network traffic into various unique categories, encompassing audio, video, text, DoS, Mirai, and bruteforce attacks.

Results from the feature importance experiment depicted in Figure 2(a), classifying benign vs. malicious traffic indicate that the top five most important attributes are Forward Maximum Packet Length (FWD Max Pkt Len), Backward Initial Window Byte Size (BWD Init Win), Flow Byte Rate (also referred as Flow Byte/s), Forward Initial Window Byte Size (FWD Init Win), and Forward Minimum Segment Size (FWD Min Seg Size). Notably, FWD Max Pkt Len and BWD Init Win present high feature importance scores, particularly in their third quartile values, implying a critical role in distinguishing benign and malicious network traffic. These features' broad range of importance values reflects their diverse influence across different classifiers and PFI runs. Moreover, the Flow Byte Rate feature shows considerable variability in its importance, as evidenced by its interquartile range. Despite not reaching the upper limit seen in the first two features, it retains a notable importance score, making it a valuable contributor to traffic classification. In contrast, the FWD Init Win feature exhibits a relatively stable and moderate range of importance values, suggesting a steady but lesser contribution to network traffic classification. Finally, while not as impactful as the top-ranking features, the FWD Min Seg Size feature still contributes to the classification task. Its median importance score, though lower, provides a meaningful addition to the overall classification task.

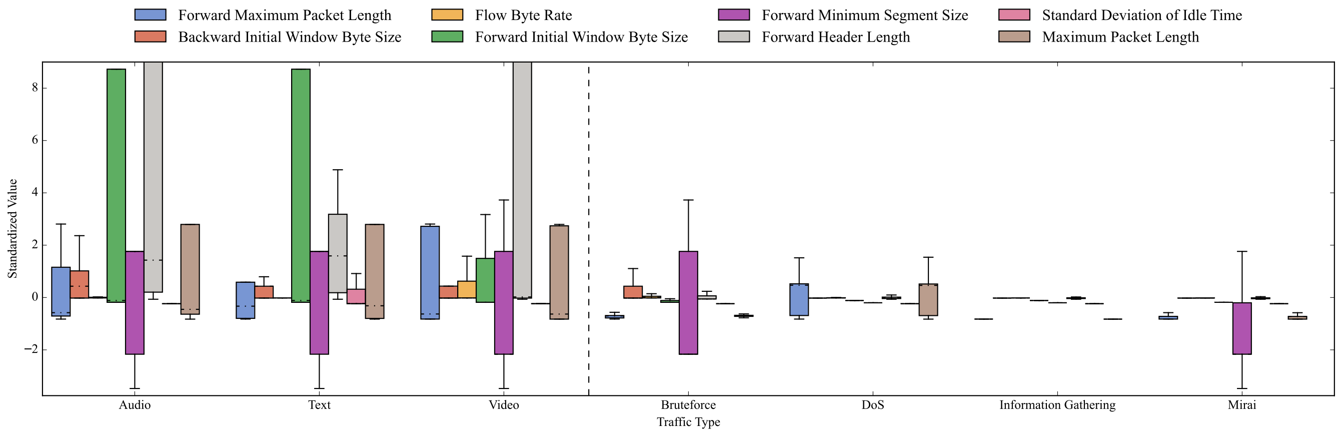


FIGURE 3. Variation of standardized feature values across traffic types. The left of the dashed line represents benign traffic (Audio, Video, Text) while the right denotes malicious types (Bruteforce, DoS, Information Gathering, Mirai).

Results from the feature importance experiment, illustrated in Figure 2(b), aimed at classifying network traffic into various unique categories, indicate that the top five most important attributes are Forward Initial Window Byte Size (FWD Init Win), Forward Maximum Packet Length (FWD Max Pkt Len), Forward Header Length (FWD Header Len), Standard Deviation of Idle Time (Std. Idle Time), and Maximum Packet Length (Max Pkt Len). The FWD Init Win feature is the most significant, supported by its nearly maximal feature importance scores across the first, second, and third quartiles. Its consistently high importance demonstrated across multiple classifiers and PFI runs, underscores its pivotal role in differentiating between various types of network traffic. Remarkably, FWD Init Win is one of the top five important features in both experiments, attesting to its relevance across distinct classification tasks. The other four features also contribute significantly to the classification task, with varying importance scores. FWD Max Pkt Len, particularly in its third quartile, substantially influences traffic classification. Additionally, FWD Header Len and Std. Idle Time plays important roles, enhancing the model's ability to distinguish between traffic types. Max Pkt Len, although not scoring as high as the others, still contributes notably to the overall classification task. These top five attributes, especially FWD Init Win featured in both experiments, play a vital role in effectively classifying network traffic.

Given the most important features identified from the feature importance analysis, we can now examine their raw values across different traffic types. Figure 3 presents the standardized feature values for the top eight most important features, allowing us to identify significant variations among the traffic types. Across the Video, Audio, and Text traffic types, we notice a notable variation in the values of the features compared to the DoS, Mirai, and Bruteforce traffic types. There seems to be a consistent pattern for the first three traffic types, where the feature values exhibit a more widespread distribution, covering a larger range of values. In contrast, the DoS, Mirai, and Bruteforce traffic types show a more concentrated distribution of feature values, with relatively lesser variations. Moreover, we can identify several features with distinct characteristics among the first three traffic types. For instance, FWD Max Pkt Len stands out with relatively high variability in the values across the Video, Audio, and Text traffic types. In contrast, features like FWD Init Win and FWD Header Len exhibit relatively stable and consistent values across the benign traffic types. We notice a different trend when examining the malicious traffic types (DoS, Mirai, and Bruteforce). The features display more uniform values, indicating less variability across these traffic types. Features such as FWD Min Seg Size and FWD Header Len show particularly distinct characteristics compared to the benign traffic types, reinforcing their relevance in distinguishing between benign and malicious traffic.

V. EXPERIMENTAL EVALUATION AND BASELINE RESULTS

In this section, we evaluate supervised and unsupervised methodologies to establish firm baseline performances for intrusion detection utilizing our dataset. This undertaking serves two functions. Firstly, it equips future research that leverages our dataset with crucial insights and performance benchmarks. Secondly, it offers robust baselines for two essential tasks in network security: supervised intrusion detection and unsupervised intrusion detection via Out-of-Distribution (OOD) detection, that is, network anomaly detection. Through this, we enable the comparison of emerging models and methodologies using our shared dataset, thereby promoting the development of more effective intrusion detection systems. Section V-B details the application of supervised methodologies to distinguish various types of network traffic while simultaneously acknowledging the inherent limitations of these methods when dealing with unseen attacks absent from the training data. In contrast, Section V-C investigates the use of unsupervised approaches for anomaly detection, emphasizing the need to incorporate a wide variety of real-world traffic patterns to boost model robustness and adaptability to changing traffic distributions.

A. DATA HANDLING AND EXPERIMENTAL DESIGN

The preprocessing phase involved removing unnecessary columns and duplicates. The columns removed were source IP and port, destination IP and port, and flow identifier, allowing us to focus on the most pertinent features for our analysis. We applied normalization and standard scaling techniques to address disparities in the scales of different features. Missing data were handled using two different strategies based on the nature of the data. Missing values in numerical data were substituted with the mean value of the respective feature. In contrast, missing values were replaced with the most frequent category for categorical data. One-hot encoding was employed specifically for the 'protocol' feature, the only categorical variable in our dataset. We refrained from performing any form of dimensionality reduction. In our preliminary experiments, we tried to balance the dataset using the Synthetic Minority Over-sampling Technique (SMOTE), specifically targeting every class with under 1,000 samples. Our goal was to ensure each class had at least 1,000 samples by synthetically generating data, and this was applied solely to the training set. However, despite these efforts, the SMOTE application did not result in any significant performance improvement. Thus, we decided not to use SMOTE in our final experiments, opting to maintain the original distribution and authenticity of the dataset.

To evaluate the models, we employed several metrics, including the F1 score, Area Under the Receiver Operating Characteristic Curve (AUROC), and Area Under the Precision-Recall Curve (AUC-PR). The F1 score balances precision and recall and provides an overall assessment of a model's accuracy. The F1 score we employed uses the macro average, the unweighted mean of the F1 scores for each class. The AUROC measures a model's capability to distinguish between classes, with a higher AUROC indicating better performance. The AUC-PR summarizes the precision-recall curve and is particularly useful in scenarios with class imbalances. These metrics were chosen based on our problem's characteristics and the need to assess the models from various perspectives.

B. BASELINES FOR SUPERVISED-BASED INTRUSION DETECTION

Our experiments for the supervised classification are carried out in three steps: (1) a binary classification to differentiate between benign and malicious traffic, (2) a multiclass classification to categorize diverse types of traffic, and (3) a multiclass classification to classify the traffic into subtypes further. In our supervised experiment, we opted for the following classifiers: RF, Decision Tree (DT), ET, Multilayer Perceptron (MLP), Support Vector Machine (SVM), and XGBoost. Although K-Nearest Neighbors was initially considered, it was later omitted from our selection due to its below-average experiment results. These models were chosen due to their widespread utilization, interpretability, and robustness in dealing with various classification problems. Refer to the appendix C-A and Table 12 for a comprehensive list of the approximated optimal parameters for each classifier.

TABLE 3. Baseline results (%) of ML models on our published dataset for supervised network intrusion detection tasks. These results provide a baseline for future research and comparison with emerging models and methodologies.

Models	Accuracy	F1 Score	AUROC	AUC-PR
Benign vs. Malicious – Binary Classification Results				
SVM	99.84	57.87	97.61	**100**
MLP	99.99	89.48	99.83	**100**
Decision Tree	**100**	96.87	97.24	**100**
Random Forest	**100**	98.01	98.62	**100**
Extra Trees	**100**	98.60	98.62	**100**
XGBoost	**100**	**98.79**	**100**	**100**
Network Traffic Types – Multiclass Classification Results				
SVM	97.73	61.66	96.45	72.44
MLP	99.94	75.60	97.81	82.62
Decision Tree	99.98	94.84	97.12	93.21
Extra Trees	99.98	96.71	99.49	97.46
Random Forest	99.98	97.28	99.53	97.66
XGBoost	**99.99**	**97.31**	**99.80**	**98.34**
Network Traffic Subtypes – Multiclass Classification Results				
MLP	99.71	78.41	99.07	85.63
SVM	99.29	80.57	97.39	81.80
Decision Tree	99.74	90.81	96.33	90.11
XGBoost	**99.79**	92.73	**99.77**	**94.45**
Random Forest	99.75	93.05	98.61	92.93
Extra Trees	99.76	**93.36**	98.77	92.95

Table 3 presents the mean performance metrics obtained from three separate runs of each method from three separate runs of each model. Binary classification results showed high performance from all models for distinguishing benign and malicious traffic, with SVM having the lowest F1 score of 57.87 (accuracy 99.84) and XGBoost having the highest F1 score of 98.79 (AUROC 100). Multiclass classification for traffic types saw similar performance, with SVM lowest and XGBoost highest (F1 score 97.31, AUROC 99.80). MLP, DT, ET, and RF exceeded 99.94 accuracies. Traffic subtype results followed this trend, with MLP and SVM lagging (F1 scores of 78.41 and 80.57, respectively) and ET leading (F1 score of 93.36).

The results demonstrate that the selected classifiers generally performed well in our dataset's binary and multiclass classifications. However, the performance was not uniform across all models in binary tasks, with SVM and MLP classifiers yielding less satisfactory F1 scores. Conversely, the XGBoost and ET classifiers excelled in all experiments, proficiently classifying benign and malicious traffic and differentiating various traffic types and subtypes. As we subdivided network traffic into more refined categories, a noticeable decline in the performance of our methods became apparent, underscoring the increased challenge in

TABLE 4. Baseline results (%) for anomaly-based intrusion detection methods. Metrics are presented for two different threshold settings: the 99th percentile and the maximum value. The table compares each model's AUROC, precision, recall, and F1 score under each threshold setting.

Models	AUC	99th Threshold			Maximum Threshold		
		Prec.	Rec.	F1	Prec.	Rec.	F1
IF	58.21	38.46	0.79	1.54	0.0	0.0	0.0
KDE	64.19	64.95	95.43	77.3	64.95	95.43	77.3
LOF	92.35	42.11	1.26	2.45	100.0	0.31	0.63
OC-SVM	96.64	99.98	57.99	73.41	99.98	9.16	16.79
Deep SVDD	**97.84**	99.98	99.68	**99.83**	99.98	99.54	**99.76**

finer-grained classifications. For a detailed understanding of the performance, refer to the classification results for each class in each experiment, provided in the Appendix (Tables 9, 10, and 11). Table 9 provides the XGBoost precision, recall, and F1 score for benign and malicious traffic. Table 10 offers details on the XGBoost precision, recall, and F1 score for each traffic *type*, whereas Table 11 delineates the Extra Trees precision, recall, and F1 score for each traffic *subtype*. This additional information enhances our understanding of the models' effectiveness across diverse traffic types and subtypes.

C. BASELINES FOR ANOMALY-BASED INTRUSION DETECTION

We formulate anomaly-based intrusion detection as an unsupervised task, conceptualizing it as an OOD detection problem. In this configuration, the in-distribution is represented by normal data, the only data type used during model training. During testing, both normal and malicious traffic are introduced, the distributions of which should ideally be separable. For this experiment, the focal evaluation metrics are the AUROC and the F1 score, computed at both the 99th percentile and maximum threshold of the scores obtained from the training set. The maximum threshold, a well-known thresholding technique, is particularly effective when the normal and malicious traffic score distributions do not overlap, thereby representing a distinct separation between these classes. Conversely, the 99th percentile threshold is employed to handle situations with extreme maximum normal scores. The anomaly detection methods selected for our unsupervised experiments include: Isolation Forest (IF), Kernel Density Estimator (KDE), Local Outlier Factor (LOF), Support Vector Machine (OC-SVM), and Deep Support Vector Data Description (Deep SVDD) [24]. For an in-depth discussion on the methods, especially details on Deep SVDD, please refer to appendix C-B and Table 12 which provides a comprehensive list of the approximated optimal parameters for each method.

Table 4 presents the mean performance metrics for each anomaly-based intrusion detection method analyzed, obtained from three separate runs of each model, all with distinct seed settings. The results indicate that the models have significant variations in their performance. For instance, the IF model struggled to distinguish between normal and anomalous traffic, resulting in the lowest AUROC of 58.21. This performance equated to a modest F1 score of 1.54 at the 99th percentile threshold. The model could not identify anomalies at the maximum threshold, yielding an F1 score of 0.0. KDE exhibited a satisfactory performance, registering an AUROC of 64.19. With an F1 score of 77.3 at both the 99th percentile and maximum thresholds, the KDE model demonstrated consistency across the two threshold settings. Following the KDE, Deep SVDD showed exceptional performance, registering the highest AUROC of 97.84 and a notable F1 score of 99.83 at the 99th percentile threshold. Deep SVDD maintained a high F1 score of 99.76 even at the highest threshold, highlighting its stable performance across both threshold settings. The performance of LOF, and OC-SVM models was inconsistent. Interestingly, the OC-SVM model showed high precision but observed a notable decrease in the recall and, therefore, the F1 score at the maximum threshold.

The results highlight the variation in model performance and emphasize the significant effect of threshold selection on said performance. This highlights the necessity for meticulous threshold selection when evaluating unsupervised anomaly detection methods. Given the intricate nature of network anomaly detection, more sophisticated strategies are commonly needed for effective anomaly identification. Take, for example, ARCADE [25], which implements a DL strategy, leveraging an adversarially regularized 1D-convolutional neural network autoencoder to learn the normal traffic pattern from raw network data. Our dataset, including raw traffic, aligns well with these advanced techniques. The strong performance of Deep SVDD in network traffic analysis further reinforces the value of adopting these advanced techniques.

VI. CONCLUSION

Addressing the widespread challenge in public network traffic datasets where there is an overrepresentation of benign and a scarcity of diverse malicious network traffic, we introduce the TII-SSRC-23 dataset available at https://kaggle.com/datasets/daniaherzalla/tii-ssrc-23. We emphasize the importance of data diversity in enhancing IDS efficacy within ML-based paradigms. TII-SSRC-23 dataset encompasses a wide spectrum of benign and malicious traffic patterns, including 32 benign and malicious traffic subtypes with 26 unique attacks launched, each enriched with many variations in traffic parameters. Although the imbalance towards malicious samples of our dataset may appear to be a drawback, we highlight that this reflects the diversity present in the malicious traffic. As previously mentioned, the representation of benign examples can be enriched with traffic from the aforementioned public datasets. By exploring feature importance analysis, we have successfully unearthed the generated data's inherent statistical tendencies and intricacies. Moreover, our experimental evaluations established benchmark performance for each subtype. These benchmarks not only serve as a baseline for upcoming research but also underscore the importance of using both supervised and unsupervised methodologies in ensuring comprehensive security coverage against a wide array of network threats.

VII. FUTURE WORK

Future improvements upon our research could benefit from expanding the TII-SSRC-23 dataset by merging it with other benign datasets, amplifying the diversity of benign traffic types, and enhancing the dataset's representativeness. Furthermore, the performance of IDS models trained on our data could be rigorously tested in real-world deployment scenarios to assess their effectiveness under real-world operating conditions. The insights from this paper can steer future research towards prioritizing traffic diversity to capture the complexities of network traffic, thereby strengthening the development of intrusion detection systems to address evolving network security challenges effectively.

APPENDIX A
PROBLEM NOTATION

Let us formalize the concepts of unidirectional and bidirectional network flows. Consider a network where packets are transmitted between different endpoints. A packet p can be defined as a tuple $p = (s_{\text{ip}}, s_{\text{prt}}, d_{\text{ip}}, d_{\text{prt}}, \tau)$, where s_{ip} is the source IP address, s_{prt} is the source port, d_{ip} is the destination IP address, d_{prt} is the destination port, and τ is the transport-level protocol used. The arrival of each packet is indicated by its corresponding timestamp t.

A. UNIDIRECTIONAL NETWORK FLOW

A unidirectional network flow $\mathcal{F} = (p_1, p_2, \ldots, p_n)$, commonly referred to as network flow, represents a sequence of n packets that share the same 5-tuple, i.e., for any pair of packets p_i and p_j, where $i, j \in \{1, 2, \ldots, n\}$, we have that $p_i = p_j$. Additionally, the packets within the flow are ordered based on their arrival timestamps, such that $t_i < t_{i+1}$, where $i \in 1, 2, \ldots, n-1$. Here, p_i represents the i-th packet in the sequence, and t_i represents the timestamp of the i-th packet.

B. BIDIRECTIONAL NETWORK FLOW

A bidirectional network flow, or a session or conversation, exchanges network flows between two endpoints. Let $\mathcal{C}$ represent a bidirectional network flow composed of two individual network flows: $\mathcal{F}_1 = (p_1, p_2, \ldots, p_m)$ and $\mathcal{F}_2 = (q_1, q_2, \ldots, q_n)$. A session $\mathcal{C}$ is defined as a tuple $\mathcal{C} = (\mathcal{F}_1, \mathcal{F}_2)$, satisfying the following conditions: for every packet p_i in $\mathcal{F}_1$ and every packet q_j in $\mathcal{F}_2$, we have $p_i = (s_{\text{ip}}, s_{\text{prt}}, d_{\text{ip}}, d_{\text{prt}}, \tau)$ and $q_j = (d_{\text{ip}}, d_{\text{prt}}, s_{\text{ip}}, s_{\text{prt}}, \tau)$. This ensures that the network flows within the session are bidirectional, where one flow contains packets moving from the source to the destination, and the other flow contains packets moving from the destination back to the source.

TABLE 5. IDS datasets malicious traffic.

Dataset	Attacks
DARPA98	DoS, privilege escalation (R2L, U2R), probing
KDD99	DoS, privilege escalation (R2L, U2R), probing
NSL-KDD	DoS, privilege escalation (R2L, U2R), probing
Kyoto 2006+	DoS, backscatter, malware, port scans, shellcode, exploits
UNIBS	None
TUIDS	Botnet, DoS, IRC botnet DDoS, probing, coordinated port scan, U2R using bruteforce SSH
ISCX 2012	Infiltrating, HTTP DoS, IRC Botnet DDoS, SSH Bruteforce
CTU-13	Botnets (Menti, Murlo, Neris, NSIS, Rbot, Sogou, Virut)
UNSW-NB15	Backdoors, DoS, exploits, fuzzers, generic, port scans, reconnaissance, shellcode, spam, worms
DDoS 2016	DDoS (HTTP, SIDDoS, Smurf ICMP, UDP)
CICIDS 2017	Botnet (Ares), DoS/DDoS, XSS, heartbleed, infiltration, SSH bruteforce, SQL injection
CIC DoS	Application layer DoS attacks (high- and low-volume HTTP DoS)
BoT-IoT	Probing (port scan, OS fingerprinting), DoS/DDoS (HTTP, TCP, UDP), information theft (data theft, keylogging)
LATAM-DDoS-IoT	DoS and DDoS attacks (HTTP, TCP, UDP)
CIC IoT	DoS (HTTP, TCP, UDP), RTSP Bruteforce
Edge-IIoTset	DoS/DDoS (HTTP, ICMP, TCP SYN, UDP), Information Gathering (Port scanning, OS fingerprinting, Vulnerability scanning), MitM (DNS and ARP spoofing), Injection attacks (XSS, SQL injection, uploading attack), Malware (backdoor, password cracking, ransomware)
N-Baiot	Botnets (Mirai, BASHLITE)
TON-IoT	Scanning, DoS, DDoS, ransomware, backdoor, injection, XSS, password cracking, MitM
TII-SSRC-23	DoS (HTTP, ICMP, MAC, UDP, TCP SYN, TCP ACK, TCP PSH, TCP RST, TCP FIN, TCP URG, TCP ECN, TCP CWR), Information Gathering (TCP Port, UDP Port, Ping, OS, Version, Script scans), Bruteforce (DNS, FTP, HTTP, Telnet, SSH), Botnet (Mirai)

C. INTRUSION DETECTION

Based on the definitions provided for unidirectional and bidirectional network flows, we present the task of Intrusion Detection. This task involves classifying a network flow $\mathcal{F}$, whether unidirectional or bidirectional, into one of two categories: benign or malicious. For this purpose, we formally introduce the Intrusion Detection function $D : \mathcal{F} \to \{0, 1\}$ that assigns binary labels to network flows. In this mapping, an output of 0 corresponds to a benign flow, whereas an output of 1 signifies a malicious flow. Note that, in certain instances, the labels assigned may vary such that -1 represents malicious flows, and 1 represents benign flows. Function D is learned from a training dataset of network flows $\mathcal{D} = \{(\mathcal{F}_1, y_1), (\mathcal{F}_2, y_2), \ldots, (\mathcal{F}_n, y_n)\}$, where each $\mathcal{F}_i$ is a network

TABLE 6. Detailed overview of the tools, parameters, and combinations employed for the generation of benign traffic.

Traffic Type	Traffic Subtype	Tool	Parameters Varied	Combinations
Audio	Audio	Mumble	Audio message length 5% disconnection rate Network interference: low, mid, high	1
Background	Background	–	Network interference: low, mid, high	1
Text	Text	Mumble	Text message length 5% disconnection rate Network interference: low, mid, high	1
Video	HTTP RTP/TS UDP	VLC	Video resolution: 240p, 360p, 480p, 720p, 1080p Audio bitrate: 96 to 192 Video bitrate: 800 to 3500 Video scale: 0.1 to 1 Frames per second: 15 to 60 Sample rate: 8000, 11025, 22050, 44100, 48000 Video codec: MPEG-4, H-264, H-265, VP8 Audio codec: MPEG, Vorbis, Opus Multiplexer: MPEG-TS, ASF/WMV, MKV, Ogg/Ogm, Webm Network interference: low, mid, high	180 per protocol

flow, and $y_i \in \{0, 1\}$ is the associated ground truth label. The Intrusion Detection problem can also be extended to address multi-class problems. In such scenarios, the function D identifies whether a flow is benign or malicious and discerns the specific type or subtype of the traffic. Consequently, function D is defined as $D : \mathcal{F} \rightarrow \{0, 1, 2, \ldots, k\}$. In this mapping, the output k represents k distinct classes of network traffic types or subtypes. Like the binary case, function D is learned from a training dataset of network flows where each flow is linked to a label indicating its traffic type or subtype. Whether a binary or multi-class case, the optimal Intrusion Detection function accurately classifies unseen network flows, thereby contributing to identifying and mitigating potential network threats.

APPENDIX B
NETWORK TRAFFIC GENERATION DETAILS

This section delves into the finer intricacies of our traffic generation procedures, detailing the specifications for each traffic type and the parameters that underwent variation during the generation process. Tables 6, 7, and 8 together provide an extensive breakdown of the elements, including traffic types, subtypes, tools, varied parameters, and an estimated number of combinations. The "Combinations" column indicates the count of traffic variations within a specific traffic subtype. This count is approximated based on the number of traffic permutations generated using the parameters varied unique to that subtype. The Mirai botnet attack is represented in Table 8, with the "Tool" column omitted since all derived attacks are associated with the Mirai botnet. We outline the parameters manipulated for each traffic subtype and their respective values and subsequently enumerate the varied parameters in the traffic generation process.

In terms of benign traffic, as presented in Section III-C, we generated audio, background, text, and video traffic. For video traffic, we manipulated eleven parameters such as network interference levels (low, mid, high), video resolutions ranging from 240 to 1080 pixels, various audio and video bitrates, video scaling factors, frame rates, sample rates, an array of video codecs (e.g., MPEG-4, H-264), audio codecs (e.g., MPEG, Vorbis), and multiplexer types (e.g., MPEG-TS, MKV), presented in Table 6. A compatibility-maintaining mapping was designed to synchronize video, audio, and multiplexer types interactions. During the data capture phase, we performed numerous rounds of traffic generation, with a Python script employed to facilitate VLC video streaming and randomize the parameters. Specific considerations were also given to factors like audio and text traffic message length, and a deliberate 5% client disconnection rate was introduced. The intricate manipulation of these parameters across various benign traffic subtypes was designed to capture real-world benign network traffic complexities.

To provide comprehensive coverage concerning traffic diversity, we conducted an intensive examination of various parameters within malicious traffic, attempting to vary the parameters extensively to achieve maximum coverage. Using dedicated tools listed in Table 7, we systematically manipulated different types of attacks, such as DoS and Information Gathering, carefully evaluating and altering the parameters specific to each attack. For all malicious traffic capture, apart from botnet traffic, we varied the network interference to capture data in low- and high-interference environments, expanded upon in Section III-B. In DoS attacks, we explored MAC, HTTP, ICMP, TCP, and UDP subtypes whilst altering attack-related parameters. The packet size and speed of transmission, critical characteristics of DoS attacks, were also purposefully adjusted, incorporating transmission modes from Hping3, "fast", "faster", and "flood" with 10 packets per second (pps), 100 pps, and over 1000 pps respectively, and payload sizes ranging from 50 to 50,000 bytes. Furthermore,

TABLE 7. Comprehensive summary of the tools, parameters, and methods used to generate malicious traffic.

Traffic Type	Traffic Subtype	Tool	Parameters Varied	Combinations
Bruteforce	FTP DNS (Fwd, Rev) SSH Telnet	Patator, Filezilla Patator, dnsmasq Patator Patator	Network interference: low, high	1 2 1 1
	HTTP fuzz	Patator, Apache, PhpMyAdmin	Request method: GET, POST Network interference: low, high	2
DoS	MAC	macof	Network interference: low, high	1
	HTTP	GoldenEye	Request method: GET, POST, Random Number of concurrent workers: 1, 20, 50 Number of concurrent sockets: 100, 500, 1000 Network interference: low, high	27
	ICMP	Hping3	Payload size: 50, 500, 5000, 50000 bytes Speed of pkt send: fast, faster, flood Network interference: low, high	16
	UDP	Hping3	Payload size: 50, 500, 5000, 50000 bytes Speed of pkt send: fast, faster, flood Random source port Bad UDP checksum (boolean) Network interference: low, high	24
	TCP ACK TCP CWR TCP ECN TCP FIN TCP PSH TCP RST TCP SYN TCP URG	Hping3	Speed of pkt send: fast, faster, flood Random source port Payload size: 50, 500, 5000, 50000 bytes Bad TCP checksum (boolean) TCP window size: default, 50, 1000 Fake tcp data offset: 0, 5, 10 Network interference: low, high	24 per attack
Information Gathering	Port Scan	Hping3	TCP Flags: ACK, FIN, PSH, RST, SYN, URG Ports 1-65535 Network interference: low, high	6
	TCP Port Scan	Nmap	Seven Scans: Connect, FIN, Maimon, NULL, SYN/ACK, Window, Xmas Timing template: "Aggressive" Bad checksum Ports 1-65535 Send string as a payload Payload size: 0, 50, 100, 5000 Network interference: low, high	42
	UDP Port Scan	Nmap	Timing template: "Aggressive" Bad checksum Ports 1-65535 Send string as a payload Payload size: 0, 50, 100, 5000 Network interference: low, high	6
	OS detection, version detection, script scanning, and traceroute	Nmap	Random MAC address Limit OS detection to only most likely matches Guess OS instead of relying on fingerprint matching Timing template: "Normal", "Aggressive" Bad checksum Set max-rate to 2 pps Ports 1-65535 Send hexadecimal value as data payload Scan 100 most common ports Payload size: 0, 50, 100, 5000 Enable fragmented IP packets Network interference: low, high	6
	Ping Scan	Nmap	Seven Scans: ICMP echo, ICMP netmask request, ICMP timestamp request, SCTP INIT, TCP ACK, TCP SYN, UDP Timing template: "Normal", "Aggressive" Bad checksum Ports 1-65535 Send string as a payload Payload size: 0, 50, 100, 5000 Network interference: low, high	42

TABLE 8. Comprehensive summary of the tools, parameters, and methods used to generate Mirai Botnet traffic type.

Traffic Type	Traffic Subtype	Parameters Varied	Combinations
Mirai Botnet	Scanning and Bruteforce	-	1
	DDoS ACK	Payload size: 50, 500, 1000 bytes Type of service: none, 1 Random source and destination ports	3
	DDoS SYN	Flag: SYN, SYN URG, SYN PSH, SYN RST, SYN FIN, SYN ACK Type of service: none, 1 Random source and destination ports	3
	DDoS DNS	Random source port	1
	DDoS GREETH	Payload size: 50, 500, 1000 Type of service: 6, 10, 70, 200 GCIP flag (boolean)	3
	DDoS GREIP	Payload size: 50, 500, 1000 Type of service: none, 1 Random source and destination ports GCIP flag (boolean)	4
	DDoS HTTP	Request method: GET, POST Number of connections: 50, 200, 800	2
	DDoS UDP	Payload size: 50, 500, 1000 Type of service: none, 1 Random source and destination ports	3
	DDoS UDP Plain	Payload size: 50, 500, 1000 Random destination ports	2

TABLE 9. XGBoost precision, recall, and F1 score for benign and malicious traffic.

Category	Precision	Recall	F1 Score
Benign	99.59	95.67	97.59
Malicious	100.00	100.00	100.00

TABLE 10. XGBoost precision, recall, and F1 score for each traffic *type*.

	Traffic Type	Precision	Recall	F1 Score
Benign	Audio	97.30	100	98.63
	Background	100	83.33	90.91
	Text	97.22	87.50	92.11
	Video	100	95.35	97.62
Malicious	Bruteforce	99.87	99.58	99.72
	DoS	99.99	100	99.99
	Information Gathering	100	100	100
	Mirai	99.75	99.33	99.54

our approach included employing Nmap for conducting comprehensive scans, including OS detection, version detection, script scanning, and traceroute. The exploration encompassed a variety of configurations, enhancing the assessment of victim systems as listed in Table 7. Some attacks, such as the DoS MAC flood and the Bruteforce attacks, had limited variability due to the tools' constraints.

Finally, Table 8 provides nuances of parameters within the context of the Mirai Botnet attack. We executed eight distinct attack vectors, each with various manipulation parameters. A script was devised for Mirai DDoS attacks incorporating the varied parameters. Different subtypes of attacks, such as DDoS ACK and DDoS SYN, entailed specific manipulations like payload size, type of service, and random source/destination ports, culminating in multiple variations. This systematic diversification across each attack subtype contributes to our dataset's comprehensive and intricate representation of malicious network activities.

APPENDIX C HYPERPARAMETERS

A. SUPERVISED METHODS

The supervised methods selected for our supervised experiments include: RF, DT, ET, MLP, SVM, and XGBoost. Each classifier underwent a hyperparameter tuning process using grid search. The grid search resulted in the following approximated optimal hyperparameters. For RF, the maximum tree depth was found to be 'none', the minimum number of samples required to split a node was 2, and the number of estimators used was 100. For DT, the function for measuring the quality of splits was 'entropy', the maximum tree depth was 'none', the minimum number of samples required at a leaf node was 1, and the minimum number of samples required to split an internal node was 5. For ET, the function for measuring the quality of splits was 'entropy', the maximum tree depth was 'none', the minimum number of samples required to split a node was 4, and the number of estimators used was 200. For the MLP, the activation function was 'tanh', the L2 penalty (regularization term) parameter was 0.0001, the configuration for the number of neurons in the hidden layers was (64, 64), and the solver for weight optimization was 'adam'. For XGBoost, the

TABLE 11. Extra Trees precision, recall, and F1 score for each traffic *subtype*.

	Traffic Subtype	Precision	Recall	F1 Score
Benign	Audio	94.74	100.00	97.30
	Background	100.00	83.33	90.91
	Text	94.87	92.50	93.67
	Video HTTP	94.44	93.15	93.79
	Video RTP	100.00	97.14	98.55
	Video UDP	96.67	100.00	98.31
Malicious	Bruteforce DNS	100.00	100.00	100.00
	Bruteforce FTP	100.00	99.57	99.78
	Bruteforce HTTP	100.00	99.21	99.60
	Bruteforce SSH	99.37	99.75	99.55
	Bruteforce Telnet	98.02	96.51	97.26
	DoS ACK	99.41	99.44	99.43
	DoS CWR	100.00	100.00	100.00
	DoS ECN	100.00	100.00	100.00
	DoS FIN	99.49	99.47	99.48
	DoS HTTP	99.14	99.52	99.33
	DoS ICMP	100.00	100.00	100.00
	DoS MAC	100.00	100.00	100.00
	DoS PSH	99.45	99.36	99.40
	DoS RST	99.62	99.67	99.64
	DoS SYN	99.99	99.98	99.99
	DoS UDP	99.99	100.00	100.00
	DoS URG	100.00	100.00	100.00
	Information Gathering	100.00	99.99	100.00
	Mirai DDoS ACK	99.87	99.33	99.60
	Mirai DDoS DNS	99.99	99.98	99.99
	Mirai DDoS GREETH	44.44	50.00	47.06
	Mirai DDoS GREIP	27.27	30.00	28.57
	Mirai DDoS HTTP	95.95	93.61	94.77
	Mirai DDoS SYN	99.46	99.75	99.61
	Mirai DDoS UDP	58.33	50.00	53.85
	Mirai Scan and Bruteforce	97.96	98.21	98.09

TABLE 12. Optimal hyperparameters for different methods.

Supervised Methods	
RF	Max tree depth: none, Min samples for split: 2, Estimators: 100
DT	Quality function: entropy, Max tree depth: none, Min samples at leaf: 1, Min samples for split: 5
ET	Quality function: entropy, Max tree depth: none, Min samples for split: 4, Estimators: 200
MLP	Activation: tanh, L2 penalty: 0.0001, Neurons config: (64, 64), Solver: adam
XGBoost	Max tree depth: 6, Learning rate: 0.1, Subsample ratio: 1, Gradient-boosted trees: 200, Subsample ratio for columns: 0.5
SVM	Penalty parameter: 1, Kernel coefficient: scale, Function: linear
Unsupervised Methods	
OC-SVM	Kernel function: linear, γ: auto, ν: 0.1
KDE	Kernel function: Gaussian, Bandwidth: auto
IF	Estimators: 2000
LOF	Neighbors: 20, Leaf size: 30
Deep SVDD	Pytorch 2.0.1, Encoder/Decoder MLP pre-training: mean squared error, 1000 epochs. Encoder: further training to reduce distance between embeddings and center. Optimization: Adam, learning rate: 1e-4, L2 penalty: 1e-6. Architecture: 79-neuron layer (ReLU) + linear layer. Latent space: 20

maximum depth of the trees was 6, the learning rate was 0.1, the subsample ratio of the training instances was 1, the number of gradient-boosted trees was 200, and the subsample ratio of columns for each split, in each level, was 0.5. Finally, for the SVM, the penalty parameter of the error term was 1, the kernel coefficient was ‘scale’, and the function used in the algorithm was ‘linear’. The detailed results of the grid search, presenting the optimal hyperparameters for each method, are provided in Table 12.

B. UNSUPERVISED METHODS

The anomaly detection methods selected for our unsupervised experiments include: IF, KDE, LOF, OC-SVM, and Deep SVDD [24]. We conducted a grid search for each method to tune hyperparameters. The approximate optimal hyperparameters derived from this procedure are: For OC-SVM, we set the kernel function to ‘linear’, γ to ‘auto’, and ν to 0.1. We set the kernel function to Gaussian for KDE and the bandwidth to ‘auto’. For IF, we set the number of estimators to 2000. For LOF, we set the number of neighbors to 20 and the leaf size to 30. The Deep SVDD was implemented using Pytorch 2.0.1, employing an encoder and decoder MLP architecture for pre-training while minimizing the mean squared error over 1000 epochs. This approach facilitates the encoder in learning the nuances of a normal distribution. After the preliminary phase of encoder pre-training, the center point is calculated, and the decoder component is eliminated. Subsequently, the encoder undergoes further training to reduce the distance between projected embeddings and the center point. The underlying logic is that by adopting this strategy, the model will be adept at mapping normal samples closer to the central point while unable to do so as efficiently for the samples not included during the training process. The deviation between the projected and center embedding is used as a scoring metric during testing. The pre-training and training phases employed Adam optimization, with a learning rate 1e-4 and an L2 penalty of 1e-6. The encoder and decoder consist of a 79-neuron layer with ReLU activation, followed by a linear layer. The latent space has been dimensioned to 20. The detailed results of the grid search, presenting the optimal hyperparameters for each method, are provided in Table 12.

REFERENCES

[1] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, “Characterization of encrypted and VPN traffic using time-related features,” in *Proc. 2nd Int. Conf. Inf. Syst. Secur. Privacy (ICISSP)*, 2016, pp. 407–414.

[2] Massachusetts Institute of Technology. (1998). *1998 DARPA Intrusion Detection Evaluation Dataset*. [Online]. Available: https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset

[3] S. Stolfo. *KDD Cup 1999 Dataset*. Accessed: Jun. 22, 2018. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[4] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, Jul. 2009, pp. 1–6.

[5] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue, and K. Nakao, “Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation,” in *Proc. 1st Workshop Building Anal. Datasets Gathering Exp. Returns Secur.*, Apr. 2011, pp. 29–36.

[6] F. Gringoli, L. Salgarelli, M. Dusi, N. Cascarano, F. Risso, and K. C. Claffy, "GT: Picking up the truth from the ground for internet traffic," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 5, pp. 12–18, Oct. 2009.

[7] S. García, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Comput. Secur.*, vol. 45, pp. 100–123, Sep. 2014.

[8] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Towards generating real-life datasets for network intrusion detection," *IJ Netw. Secur.*, vol. 17, no. 6, pp. 683–701, 2015.

[9] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, no. 3, pp. 357–374, May 2012.

[10] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.

[11] M. Alkasassbeh, G. Al-Naymat, A. B. A. Hassanat, and M. Almseidin, "Detecting distributed denial of service attacks using data mining techniques," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 1, pp. 1–10, 2016.

[12] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. ICISSP*, 2018, pp. 108–116.

[13] H. H. Jazi, H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling," *Comput. Netw.*, vol. 121, pp. 25–36, Jul. 2017.

[14] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, A. Shabtai, D. Breitenbacher, and Y. Elovici, "N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Comput.*, vol. 17, no. 3, pp. 12–22, Jul. 2018.

[15] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Gener. Comput. Syst.*, vol. 100, pp. 779–796, Nov. 2019.

[16] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustain. Cities Soc.*, vol. 72, Sep. 2021, Art. no. 102994.

[17] S. Dadkhah, H. Mahdikhani, P. K. Danso, A. Zohourian, K. A. Truong, and A. A. Ghorbani, "Towards the development of a realistic multidimensional IoT profiling dataset," in *Proc. 19th Annu. Int. Conf. Privacy, Secur. Trust (PST)*, Aug. 2022, pp. 1–11.

[18] J. G. Almaraz-Rivera, J. A. Perez-Diaz, J. A. Cantoral-Ceballos, J. F. Botero, and L. A. Trejo, "Toward the protection of IoT networks: Introducing the LATAM-DDoS-IoT dataset," *IEEE Access*, vol. 10, pp. 106909–106920, 2022.

[19] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning," *IEEE Access*, vol. 10, pp. 40281–40306, 2022.

[20] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *Proc. IEEE Symp. Secur. Privacy*, May 2010, pp. 305–316.

[21] A. Neumann, C. Aichele, M. Lindner, and S. Wunderlich, "Better approach to mobile ad-hoc networking (BATMAN)," IETF Draft, pp. 1–24, 2008.

[22] G. Maciá-Fernández, R. A. Rodríguez-Gómez, and J. E. Díaz-Verdejo, "Defense techniques for low-rate DoS attacks against application servers," *Comput. Netw.*, vol. 54, no. 15, pp. 2711–2727, Oct. 2010.

[23] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the Mirai botnet," in *Proc. 26th USENIX Secur. Symp. (USENIX Security)*, 2017, pp. 1093–1110.

[24] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4393–4402.

[25] W. T. Lunardi, M. A. Lopez, and J.-P. Giacalone, "ARCADE: Adversarially regularized convolutional autoencoder for network anomaly detection," *IEEE Trans. Netw. Service Manage.*, vol. 20, no. 2, pp. 1305–1318, Jun. 2023.

DANIA HERZALLA received the B.Sc. degree in computer science from New York University, Abu Dhabi. She is an Associate Security Engineer at the Secure Systems Research Center, Technology Innovation Institute, Abu Dhabi, UAE. Her work has been centered around network intrusion detection and jamming detection research.

WILLIAN TESSARO LUNARDI (Member, IEEE) received the Ph.D. degree in computer science from the University of Luxembourg. He is currently a Lead Researcher with the Technology Innovation Institute, Abu Dhabi, United Arab Emirates, where he has been with the forefront of contrastive learning methods and adversarial strategies for out-of-distribution detection. Prior to this position, he was a Research Associate with the University of Luxembourg, where his research primarily encompassed neural combinatorial optimization, preventive maintenance, and combinatorial optimization. Throughout his career, he has contributed to more than 30 international journals and conferences. His research interests include deep learning, self-supervised learning, out-of-distribution detection, and combinatorial optimization.

MARTIN ANDREONI (Member, IEEE) received the bachelor's degree in electronic engineering from Universidad Nacional de San Juan (UNSJ), Argentina, in 2011, the master's degree in electrical engineering from the Federal University of Rio de Janeiro (COPPE/UFRJ), in 2014, and the Ph.D. degree from the Teleinformatics and Automation Group (GTA), COPPE/UFRJ, and the Phare Team of Laboratoire d'Informatique de Paris VI (LIP6), Sorbonne Université, France, in 2018. He was a Researcher with the Samsung Research and Development Institute, Brazil. He is currently a Principal Wireless Security Researcher with the Secure System Research Center, Technology Innovation Institute, Abu Dhabi, United Arab Emirates. He has coauthored more than 50 publications and patents in security, virtualization, traffic analysis, and big data.