ToN_IoT: The Role of Heterogeneity and the Need for Standardization of Features and Attack Types in IoT Network Intrusion Data Sets

Article *in* IEEE Internet of Things Journal · May 2021

DOI: 10.1109/JIOT.2021.3085194

Tim Marijn Booij
TNO

Irina Chiscop
TNO

Erik Meeuwissen
TNO

Nour Moustafa
UNSW Canberra

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3085194, IEEE Internet of Things Journal

ToN_IoT: The Role of Heterogeneity and the Need for Standardization of Features and Attack Types in IoT Network Intrusion Datasets

Tim M. Booij*, Irina Chiscop*, Erik Meeuwissen*, Nour Moustafa†, Frank T. H. den Hartog† *Netherlands Organisation for Applied Scientific Research, TNO
{tim.booij, irina.chiscop, erik.meeuwissen}@tno.nl †University of New South Wales, Canberra, Australia
{nour.moustafa, frank.den.hartog}@unsw.edu.au

***Abstract*—The Internet of Things (IoT) is reshaping our connected world as the number of lightweight devices connected to the Internet is rapidly growing. Therefore, high-quality research on intrusion detection in the IoT domain is essential. To this end, network intrusion datasets are fundamental, as many attack detection strategies have to be trained and evaluated using such datasets. In this paper, we introduce the description, statistical analysis, and machine learning evaluation of the novel ToN_IoT dataset. Comparison to other recent IoT datasets shows the importance of heterogeneity within these datasets, and how differences between datasets may have a huge impact on detection performance. In a cross-training experiment, we show that the inclusion of different data collection methods and a large diversity of the monitored features are of crucial importance for IoT network intrusion datasets to be useful for the industry. We also explain that the practical application of IoT datasets in operational environments requires the standardization of feature descriptions and cyberattack classes. This can only be achieved with a joint effort from the research community.**

***Index Terms*—Intrusion detection, Internet of Things, statistical analysis, Machine learning algorithms, Network security**

I. Introduction

A. Background

The number of devices connected to the Internet is growing rapidly - and ”things”, or Internet of Things (IoT) devices play a big role in this. Due to technological progress and the ease with which smart devices can be produced, the IoT market is constantly growing with new IoT devices being connected to the Internet every day [1]. This has led to a rapid development of IoT applications across different fields of industry, such as automotive [2], smart home systems [3], space applications [4], healthcare [5], manufacturing and retail [6]. This also raises many new security challenges. The 2020 IoT Threat Report [7] by the global threat intelligence team at Palo Alto Networks, shows that 98% of IoT data is unencrypted, possibly exposing confidential information, and that IoT devices often run outdated software versions, becoming vulnerable to exploits. According to this report, exploiting device vulnerabilities is the most encountered IoT threat, followed by malware and poor user practices such as re-using passwords. For a complete taxonomy of IoT-related cyber threats, we refer to the survey in [8].

There is a growing body of literature recognizing the importance of threats in IoT, focusing on developing advanced intrusion detection techniques. These approaches include novel signature-based, anomaly-based, and specification-based models. Validating the correctness of these new detection models remains a challenge though, which is largely due to a lack of readily available high-quality labeled datasets [9]. The main difficulty in creating such datasets stems from a) the fact that typical IoT devices generate significantly less traffic compared to work stations and servers in regular networks, and b) the continuous deployment of new ”things” in the Internet, meaning that new types of traffic and attacks are continuously being created. The latter puts requirements to the heterogeneity of the datasets, i.e. they need to contain many different IoT devices and related types of traffic and attacks. Only heterogeneous datasets enable security researchers to distinguish specifically between benign and malicious traffic originating from IoT devices, and are therefore an extremely valuable asset when employing anomaly detection strategies.

B. Motivation and research contribution

The survey in [10] shows that most of the current network intrusion detection methods aimed at IoT systems are assessed and evaluated by emulating or simulating attacks using relatively old datasets such as KDD-99 [11], NSL-KDD, UNSW-NB15 [12], or CICIDS [13]. These datasets do not sufficiently represent the heterogeneous nature of current IoT networks, which typically include many different protocols, standards and technologies. A few recently developed dedicated IoT datasets such as IoT Network Intrusion Dataset [14], Aposemat IoT-23 [15] and N-BaIoT [16] may be able to address this shortcoming, and provide a benchmark for efficient comparison between different detection methods. However, as we will illustrate later in this article, they are still fairly limited in the number of IoT devices and diversity of attacks they consider. Moreover, these datasets have often been created with only a single data source, for instance network data in the form of pcap files (in the case of IoT Network Intrusion Dataset) or sensor measurement data (in the case of N-BaIoT).

With this article we want to demonstrate the relevance of dataset heterogeneity for effective intrusion detection in IoT, and show how such heterogeneity improves the learning rate of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3085194, IEEE Internet of Things Journal

machine learning based detection algorithms. We will do this by presenting and analyzing the novel ToN_IoT dataset. It is the first IoT network intrusion dataset to combine information from four heterogeneous data sources (pcap files, Bro logs, sensor data and OS logs), and particular attention has been given to include many different devices and attack types. Our research contributions are:

- We provide a full statistical description of the dataset, a qualitative analysis, and a comparison with the most relevant other dedicated IoT datasets.
- We investigate the importance of dataset heterogeneity, and demonstrate through numerical experiments how this heterogeneity indeed improves the learning rate of machine learning based detection algorithms.
- We show, by means of a novel way of cross-training, that the need for heterogeneity automatically leads to a need for international standardization of the names and definitions of features and attack types. We have identified this as a blind spot in the current academic research of intelligent IoT network intrusion detection, and this lack of standardization may restrict the further adoption of intelligent IoT intrusion detection by the industry.

C. Structure of the paper

The remainder of the paper is structured as follows. In Section II, an overview is presented of publicly available IoT datasets. In Section III, the ToN_IoT dataset is introduced, explaining the data collection set-up and the obtained features. In Section IV, various statistically descriptive properties of the ToN_IoT are provided, which is followed by a discussion of feature correlations in Section V. Various classifiers are then tested on ToN_IoT as well as Aposemat IoT-23, and the results are explained in Section VI. Concluding remarks and directions for future work are formulated in Section VII.

II. Overview of publicly available IoT datasets

A. Measuring IoT data

Most IoT devices inherently do not have a large footprint regarding network traffic. It primarily consists of sensor data being sent from the devices, for example changes in temperature. It also contains log and raw data. These three different data types are described here.

1) **Sensor data** - IoT devices typically contain sensors, functional software and network connectivity functionality. Smart watches, door locks, temperature regulators or voice controllers - each of these devices produce some kind of sensory data which is communicated over a network to, for example, a regulating application. Sensor data may reveal information about the (mis)use of the smart device, making it imperative to be tracked.
2) **Raw data** - Besides sensor data, every data packet sent through a network by an IoT device contains additional information, for instance about connectivity and network protocols. This provides raw information about the internal and external activity of these IoT devices, and therefore can complement sensor data.
3) **Log data** - Raw pcap data files collected from network analyzers can get very large and tedious to analyze. Therefore, special software such as Zeek (f.k.a. Bro), designed to unobtrusively observe network traffic, interpret these observations, and create compact transaction logs containing the results, is widely used by analysts in security contexts today.

Heterogeneity is an important property of a modern IoT network intrusion detection dataset for validating models. With heterogeneity we mean that the network used to create the dataset should contain many different types of devices and network technologies, and the intrusion detection dataset should reflect this heterogeneity in its diversity of traffic data as well as attack types. Heterogeneity is important, because the rapid growth of IoT in recent years is partly due to the procreation of various new technologies that simplify the installation and use of networked devices, which has led to a wide variety of standards and solutions at each network layer and in each application stack.

To formulate the data heterogeneity of ToN_IoT, the datasets have $D = \{d_1, d_2, ..., d_n\}$ data sources, each of which contains $X = \{x_1, x_2, ..., x_m\}$ multivariate variables (i.e., attributes). In our study, D and its X were gathered from disparate data sources, involving d_1 (network traffic), d_2 (telemetry data of IoT systems), d_3 (audit traces of Linux OS), and d_4 (audit traces of Window OS). Each data source, has X attributes that belong to ($X \in R \cup C$), where R and C denote the entire numeric and nominal values respectively, of attributes extracted from the layers of the TCP/IP stack. It is essential to estimate how these heterogeneous values can be flexibly fitted under the same multivariate or mixture distribution model (e.g., a Gaussian mixture model). This will help to distinguish between suspicious and legitimate observations.

B. IoT datasets for network intrusion detection

Automation of intrusion detection has been an important topic of research since the emergence of digital networks, and it is widely accepted that datasets containing both actual attacks and benign traffic are required. With these datasets, security researchers and analysts can, for example, use machine learning techniques to evaluate the differences between benign and malicious traffic patterns. Many such datasets are already publicly available. But most of these consider generic network traffic (e.g. NSL-KDD [11] and CTU-13 [17]), and only very few are designed specifically for anomaly detection in IoT networks.

Only recently, a few datasets have been released that were specifically designed for intrusion detection in IoT networks, all with their own specific goals and data types. Here, we will elaborate on these datasets, with a particular focus on their heterogeneity. The following datasets are taken into account, as they are relatively recent and display at least some level of heterogeneity: DS2OS traffic traces [18], Bot-IoT [19], IoT Network Intrusion Dataset [14], N-BaIoT [16] & Aposemat IoT-23 [15].

The DS2OS traffic traces dataset was introduced in [18] and is publicly available on Kaggle [20]. It consists of synthetic data generated in a virtual IoT environment by means of a Distributed Smart Space Orchestration System (DS2OS). The system architecture of the IoT environment contains a set of microservices which communicate with each other using the Message Queuing Telemetry Transport (MQTT) protocol. The dataset contains 13 different features obtained by monitoring connections between 7 different Virtual State Layer (VSL) service types that connect illumination controllers, movement sensors, thermostats, solar batteries, washing machines, door locks, and smart phones. Although it includes information about the microservice source, destination, operation and so forth, this dataset does not contain any NetFlow or packet data. It only contains features which were specifically designed for detecting anomalies in the IoT traffic frequencies and the communication baseline models of the microservices. Therefore, this dataset can only be used for a small class of detection purposes, and will be discarded in the remaining analysis of this paper.

The Bot-IoT dataset [19] was developed to facilitate botnet identification in IoT networks. An IoT network was simulated using virtual machines (both normal and attacking). Kali VMs performed port scanning, DDoS and other Botnet attacks. On Ubuntu VMs, the Node-red tool was used for simulating various IoT sensors including a weather station, a smart fridge, motion activated lights, a garage door and a thermostat. A total of 46 network features were extracted using the Argus tool. No sensor or log information was recorded. The pcap files of the Bot-IoT dataset are accessible for download at [21]. Bot-IoT is the only dataset which collects data using simulated IoT devices. The fact that no real IoT hardware is monitored makes the dataset less representative of real IoT traffic. This has triggered some of the original creators of the BoT-IoT dataset to create the ToN_IoT dataset as discussed in this paper. Given the deficiencies of DS2OS and BoT-IoT as described above, the remainder of the paper will only focus on the comparison of ToN_IoT with the IoT Network Intrusion Dataset, N-BaIoT and Aposemat IoT-23.

IoT Network Intrusion Dataset

The IoT Network Intrusion Dataset [14], published in 2019, was created using two typical smart home devices, a SKT NUGU (NU 100) and EZVIZ Wi-Fi Camera (C2C Mini O Plus 1080P), a few laptops and a few smartphones. These devices were all connected to the same wireless network on which different attacks were simulated using tools such as Nmap. At the time of its publication, it was one of the first datasets aimed specifically at intrusion detection in IoT environments. The dataset contains 42 raw network packet files (pcap) collected at different points in time. No other types of data were collected, which means that the dataset does not qualify as heterogeneous given our earlier definition.

N-BaIoT

The work presented in [16] introduces the N-BaIoT dataset, which has been specifically designed for network-based botnet detection. The IoT devices were connected via Wi-Fi to several access points, and wired to a central switch further linked to a router. Two types of botnet attacks were carried out: BASHLITE (executed using the binaries of IoTPOT [22]) and Mirai (executed using the publicly available source code [23]). Unlike the IoT Network Intrusion Dataset, the features of N-BaIoT are statistical quantities derived from sensor logs collected from each of the devices in the experimental set-up. Given the creators' focus on profiling device behaviour, the available data is separately provided for each device.

Aposemat IoT-23

The most recent dataset published on the topic of IoT traffic and network monitoring is Aposemat IoT-23 [15]. Aposemat IoT-23's IoT traffic is captured from real hardware IoT devices (a Philips HUE smart LED lamp, an Amazon Echo home intelligent personal assistant and a Somfy Smart Door Lock). Data was collected within the Stratosphere project and includes 20 captures of malware executed in IoT devices, and 3 captures of benign IoT devices traffic. The executed attacks include different botnets such as Mirai and Torii. Besides the original packet capture files, netflows generated by Zeek/Bro IDS and their labels are also available.

III. Description of the new ToN_IoT dataset

In this paper, we will analyze a new dataset, ToN_IoT, and provide a first qualitative comparison between this and other IoT network security datasets. ToN_IoT is a collection of datasets [24] that are a new generation of IoT and Industrial IoT (IIoT) datasets for evaluating the fidelity and efficiency of different cybersecurity applications based on Artificial Intelligence (AI). They are aimed at applications such as intrusion detection, threat intelligence, adversarial machine learning, and privacy-preserving models.

The datasets were called 'ToN_IoT' as they include heterogeneous data sources collected from **T**elemetry datasets of IoT and IIoT sensors, **O**perating systems datasets of Windows 7 and 10 as well as Ubuntu 14 and 18 TLS, and **N**etwork traffic datasets. The datasets were collected from a realistic and large-scale testbed network designed at the IoT Lab of UNSW Canberra Cyber, connecting many virtual machines, physical systems, hacking platforms, cloud and fog platforms, and IoT sensors to mimic the complexity and scalability of IIoT and Industry 4.0 networks.

The testbed architecture contains three layers: edge/IoT, fog and cloud, as depicted in Figure 1. The edge and fog layers are similar in their offering of on-premise services like cloud services, but they allow the handling of large-scale data sources, and the application of data analytics and intelligence near the end users. The key difference between the edge and fog layers is that the edge layer places intelligence, analytics, and processing power in devices, such as embedded automation controllers and lightweight IoT devices, while the fog layer places intelligence, analytics, and computing power somewhere else in the Local Area Networks (LANs). For a complete elaboration of this testbed, we refer to the work done in [25] and [26], but in this paper, each of the three layers will be briefly explained in the following paragraphs.

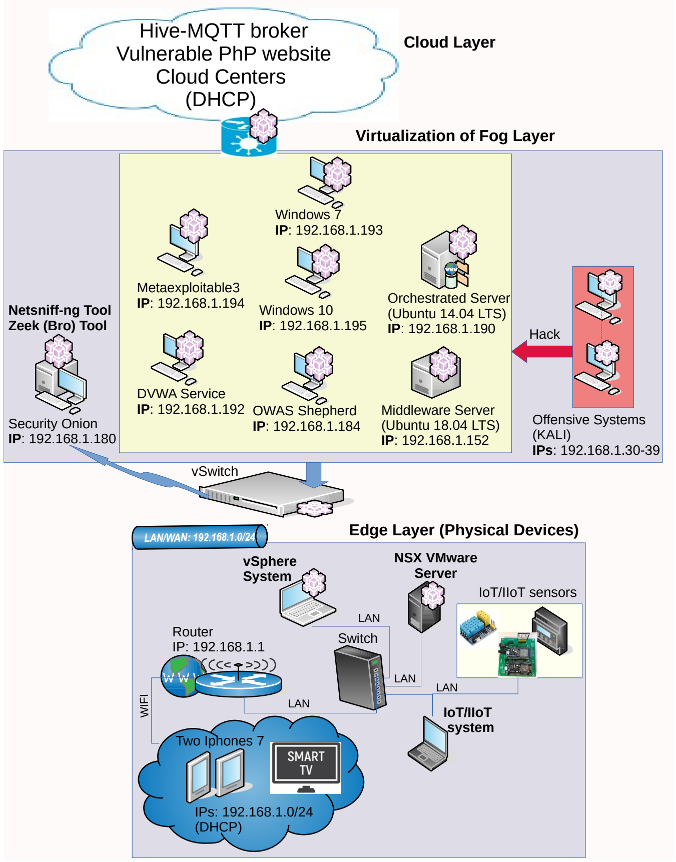

Fig. 1. Testbed architecture of ToN_IoT datasets for collecting network traffic

Edge layer

The edge layer involves the physical devices and their operating systems. They form the infrastructure on which the fog layer and cloud layer is deployed by means of virtualization. It contains various IoT/IIoT devices, including weather and light bulb sensors, smartphones and smart TVs, as well as host systems such as workstations and servers (employed to link with IoT/IIoT devices), and physical gateways (i.e., routers and switches) to the Internet. The NSX-VMware hypervisor platform was installed on a host server at the edge layer to manage and control the Virtual Machines (VMs) created at the fog layer.

Fog layer

The fog layer contains the virtualization technology that programs and controls the VMs and their services using the NSX-VMware platform. It includes the VMs of client systems (i.e., Windows 7 and 10) and vulnerable systems (i.e., Metaspoitable3, OWASP security Shepherd, and Damn Vulnerable Web App (DVWA)). It also involves offensive systems (Kali Linux VMs) and scripts of hacking scenarios that were employed to breach vulnerable systems in the testbed. An Ubuntu 18.04 Middleware server was also deployed. The IoT/IIoT services were managed with this server using the Node-red and Mosquitee MQTT broker tools to operate seven IoT/IIoT sensors: weather, smart garage door, smart fridge, smart TCP/IP Modbus, GPS tracker, motion-enabled light, and smart thermostat. Furthermore, an Ubuntu 14.04 LTS orchestrated server was deployed to offer network services, including DNS (i.e., mydns.com), HTTP(s), DHCP, email server (i.e., Zimbra), Kerberos, and FTP, and to generate network traffic between VMs using the Ostinato traffic generator.

Cloud layer

The cloud layer involves the cloud services configured online in the testbed. The fog and edge services are connected with the public HIVE MQTT dashboard, public PHP vulnerable website, cloud virtualization services (e.g., Microsoft Azure or AWS), and cloud data analytics services. The public HIVE MQTT dashboard allows publishing and subscribing to the sensing data of the IoT/IIoT services using the Node-red tool. The public PHP vulnerable website was employed to execute injection attacking events. The other cloud services were configured either in Microsoft Azure or AWS to transfer sensing data to the cloud and visualize their behaviors.

A. Threat model

A threat model has been utilized as described by Alsaedi et al. [26] to simulate realistic attacks for the dataset. These attacks are then monitored, and observations are labeled as either *normal* or *attack*, and which type of attack is performed. The following nine attack families were utilized in the datasets:

1) **Scanning attack** - Nessus [27] and the Kali Linux Nmap tools were executed against the target subnet 192.168.1.0/24 and all other public vulnerable systems such as the Public MQTT broker and the vulnerable PHP website.
2) **Denial of Service (DoS) attack** - DoS attack scenarios, created with Python scripts using the Scapy package, were utilized against any vulnerable element in the IoT testbed network.
3) **Distributed Denial of Service (DDoS) attack** – DDoS attacks were executed against several nodes, using Python scripts and the ufonet toolkit [28].
4) **Ransomware attack** - Ransomware attacks were executed against targets running Windows OS that are monitoring IoT services, by means of Metasploit exploiting Server Message Block vulnerabilities.
5) **Backdoor attack** – The offensive systems with IP addresses 192.168.1.33,37 keep persistence in compromised machines by Metasploit executing a bash script with the command ”runpersistence-h”.
6) **Injection attack** - Various offensive systems inject data to web applications of Damn Vulnerable Web Application (DVWA) and Security Shepherd VMs, and to webpages of other IoT services. These attacks include SQL injection, client-side injection, broken authentication and data management, and unintended data leakage.
7) **Cross-site Scripting (XSS) attack** - XSS code was injected to the same machines and services as targeted with Injection attacks, by means of malicious bash scripts of Python code using the XSSer toolkit [29].

8) **Password attack** – Kali Linux Hydra and Cewl toolkits were used. They were configured by means of automated bash script to concurrently launch password hacking attacks against any vulnerable nodes in the testbed.
9) **Man-In-The-Middle (MITM) attack** – Kali Linux Ettercap was employed to execute ARP spoofing, ICMP redirection, port stealing and DHCP spoofing attacks.

B. Features

An overview of the ToN_IoT data features is shown in Tables I-VI whilst the data labels are given in Table VII. These tables will elaborate on and provide a description for each feature in the datasets.

TABLE I
TON_IOT CONNECTION ACTIVITY FEATURES

Service: Connection activity		
ID	Feature	Description
1	*ts*	Timestamp of connection between flow identifiers
2	*src_ip*	Source IP addresses which originate endpoints’ IP addresses
3	*src_port*	Source port which originate endpoint’s TCP/UDP ports
4	*dst_ip*	Destination IP addresses which respond to endpoint’s IP addresses
5	*dst_port*	Destination ports which respond to endpoint’s TCP/UDP ports
6	*proto*	Transport layer protocols of flow connections
7	*service*	Dynamically detected protocols, such as DNS, HTTP and SSL
8	*duration*	The time of the packet connections, which is estimated by subtracting ’time of last packet seen’ and ’time of first packet seen’
9	*src_bytes*	Source bytes which are originated payload bytes of TCP sequence numbers
10	*dst_bytes*	Destination bytes which are responded payload bytes from TCP sequence numbers
11	*conn_state*	Various connection states, such as S0 (connection without replay), S1 (connection established), and REJ (connection rejected)
12	*missed_bytes*	Number of missing bytes in content gaps

TABLE II
TON_IOT STATISTICAL ACTIVITY FEATURES

Service: Statistical activity		
ID	Feature	Description
13	*src_pkts*	Number of original packets which is estimated from source systems
14	*src_ip_bytes*	Number of original IP bytes which is the total length of IP header field of source systems
15	*dst_pkts*	Number of destination packets which is estimated from destination systems
16	*dst_ip_bytes*	Number of destination IP bytes which is the total length of IP header field of destination systems

TABLE III
TON_IOT DNS ACTIVITY FEATURES

Service: DNS activity		
ID	Feature	Description
17	*dns_query*	Domain name subjects of the DNS queries
18	*dns_qclass*	Value which specifies the DNS query classes
19	*dns_qtype*	value which specifies the DNS query types
20	*dns_rcode*	Response code values in the DNS response
21	*dns_AA*	Authoritative answers of DNS, where T denotes server is authoritative for query
22	*dns_RD*	Recursion desired of DNS, where T denotes request recursive lookup of query
23	*dns_RA*	Recursion available of DNS, where T denotes server supports recursive queries
24	*dns_rejected*	DNS rejection, where DNS queries are rejected by the server

TABLE IV
TON_IOT SSL ACTIVITY FEATURES

Service: SSL activity		
ID	Feature	Description
25	*ssl_version*	SSL version which is offered by the server
26	*ssl_cipher*	SSL cipher suite which the server chose
27	*ssl_resumed*	SSL flag indicates the session that can be used to initiate new connections, where T refers to the SSL connection is initiated
28	*ssl_established*	SSL flag indicates establishing connection between two parties, where T refers to establishing the connection
29	*ssl_subject*	Subject of the X.509 cert offered by the server
30	*ssl_issuer*	Trusted owner/originator of the SLL and digital certificate (certificate authority)

TABLE V
TON_IOT HTTP ACTIVITY FEATURES

Service: HTTP activity		
ID	Feature	Description
31	*http_trans_depth*	Pipelined depth into the HTTP connection
32	*http_method*	HTTP request methods such as GET, POST and HEAD
33	*http_uri*	URIs used in the HTTP request
34	*http_version*	The HTTP version utilised such as V1.1
35	*http_request_body_len*	Actual uncompressed content sizes of the data transferred from the HTTP client
36	*http_response_body_len*	Actual uncompressed content sizes of the data transferred from the HTTP server
37	*http_status_code*	Status codes returned by the HTTP server
38	*http_user_agent*	Values of the User-AGent header in the HTTP protocol
39	*http_orig_mime_types*	Ordered vectors of mime types from source system in the HTTP protocol
40	*http_resp_mime_types*	Ordered version of mime types from destination system in the HTTP protocol

C. Qualitative comparison of datasets

To understand the applicability and limitations of the datasets discussed in this paper, Table VIII shows a qualitative comparison between the most representative datasets in our overview. The first striking conclusion is that ToN_IoT is the only dataset to combine four different data types: packet capture, Bro logs, sensor data and OS logs. The Aposemat IoT-23 dataset is the most similar in that respect, including pcaps and Bro logs. The ToN_IoT dataset also has the highest number of different IoT devices being monitored. In terms of size, the Aposemat IoT-23 dataset is clearly the largest, being an order of magnitude larger than ToN_IoT. N-BaIoT is the only dataset that does not include any kind of packet capture, which is why further on

TABLE VI
TON_IOT VIOLATION ACTIVITY FEATURES

Service: Violation activity		
ID	Feature	Description
41	*weird_name*	Names of anomalies/violations related to protocols that happened
42	*weird_addl*	Additional information is associated to protocol anomalies/violations
43	*weird_notice*	It indicates if the violation/anomaly was turned into a notice

TABLE VII
TON_IOT DATA LABELS

Service: Data labeling		
ID	Feature	Description
44	*label*	Tag normal and attack records, where 0 indicates normal and 1 indicates attacks
45	*type*	Tag attack categories, such as normal, DoS, DDoS, and backdoor attacks, and normal records

in this paper we will only focus on the other three datasets in the context of IoT network intrusion detection. Of those three datasets, ToN_IoT has the highest number of features.

Regarding the diversity of simulated and executed malware, the ToN_IoT dataset appears to have fewer attack classes than the IoT Network Intrusion Dataset and Aposemat IoT-23. But here we are confronted with the fact that how to distinguish attack classes is not well standardized within the community. For instance, what one would consider different attack subclasses, someone else might count as individual categories. For example, the C&C-FileDownload and C&C-HeartBeat attacks in Aposemat IoT-23 could easily be seen as just one technique. As shown in Table IX, the ToN_IoT dataset includes attacks which are not seen in the other two datasets. This lack of standardization is a serious hindrance in the further analysis and practical use of IoT security datasets, as we will evidence later in this paper.

From Table VIII, we conclude that ToN_IoT and Aposemat IoT-23 are currently the most representative datasets for IoT network traffic intrusion detection, because they both show a large number of records, combine two or more data sources, contain information about modern attacks (which are actually executed instead of simulated), and are very recent. These two datasets will be compared in more depth regarding their detection performance for various algorithms. The hypothesis is that intrusion detection models trained using ToN_IoT will perform better than those using Aposemat IoT-23 due to the higher level of heterogeneity that ToN_IoT exhibits, i.e. we hypothesize that ToN_IoT better captures the diverse characteristics of IoT network traffic. Before diving into this comparison, we will take a closer look at the characteristics of the ToN_IoT dataset.

IV. STATISTICAL DESCRIPTIVE OBSERVATIONS

Table X shows how training and testing sets have been created from ToN_IoT. For every category listed in Table X, the ToN_IoT dataset has been randomly split into two datasets with ratio 60%:40%, with the first being allocated to the training set (TR) and the second to the test set (TS). In this section, we provide a statistical analysis and comparison of the training and testing set. All analyses in this section and in Section VI have been performed in Python using the *sklearn* package. The scripts used throughout this paper can be found in [30] and all significant symbols can be found in Table XI.

A *z-score* transformation is used for scaling the data values and asserting that there is no bias to ward large numbers. The z-score Z is given by eq 1:

$$Z = \frac{x - \mu}{\sigma} \tag{1}$$

such that x is an attribute in the entire set $X = \{x_1, x_2, ..., x_m\}$. μ and σ refer to the mean and standard deviation of each attribute, respectively. This transformation standardizes every x to have $\mu = 0$ and $\sigma = 1$ in the entire X. The sign of Z illustrates if the value of x is below or above μ and σ.

After normalizing the data attributes, a Kolmogorov-Smirnov test is used to examine which data distribution fits the nature of the data best (i.e., its normality and linearity) to decide which machine learning model can train and validate data with the lowest noise and false alarm rate. This test estimates the distance between the empirical distribution function of the attributes $F(x) = x$, for $0 \leq x \leq 1$ and the Cumulative Distribution Function (CDF) of $f(x)$, i.e., $Sn(x) = rand(x/N)$, where $rand(x) \subset [0, 1]$, and N is the number of random numbers ($rand(x)$). The test estimates whether an attribute (x) achieves the null hypothesis (H_0) of a normal/uniform distribution or not (i.e., the alternative hypothesis (H_1)) via calculating:

$$D = |F(x) - Sn(x)| \tag{2}$$

The output D is compared with D_{alpha}, where $alpha$ denotes the level of significance, which estimates the probability of rejecting H_0, when $p\text{-}value = 1$ for the uniform or normal distribution.

Fig. 2 shows the probability distribution of the features in both sets, using the one sampled Kolmogorov-Smirnov test against a CDF distribution. Apparently, the distribution of features is nonlinear and non-normal. The two curves are almost identical, and most features attain probability values in the interval 0.45-0.5, showing a fairly good fit. Exceptions are features such as *connection_state*, *proto* and *service*. All non-numerical features are mapped to numerical representations in the processing step, which may cause lower probability values. To measure to what extent the attributes (x) are skewed from a normal distribution, the skewness is computed by:

$$Skewness = \frac{\sum_{i=1}^{n}(x_i - \mu)^3}{n\sigma^3} \tag{3}$$

where n is the number of records in an attribute.

The asymmetry between the training and the testing set is described by the multivariate skewness function, shown in Fig. 3. Here, all features except *connection_state* are positive, which means that the majority of the features are on the right-hand side of the probability density function distribution.

TABLE VIII
COMPARISON OF DATASETS FOCUSED ON IOT ATTACKS

	IoT Network Intrusion Dataset	**Aposemat IoT-23**	**N-BaIoT**	**ToN_IoT**
Dataset size	3M packets	325M log entries	7M sensor log entries	22M log entries
% malicious/benign	42%/58%	90%/10%	92%/8%	96,4%/3,6%
Number of attack classes	10	15	10	9
Number of features	# from .pcap	22	115	46
IoT device types	2	3	4	9
Data type	Raw	Raw & Log	Sensor	Raw, Log & Sensor
Year of publication	2019	2020	2018	2020

TABLE IX
DISTRIBUTION OF ATTACK CATEGORIES AMONGST NOVEL IOT DATASETS

	IoT Network Intrusion	Aposemat IoT-23	ToN_IoT
Attack category	**# of logs**	**# of logs**	**# of logs**
Backdoor	-	-	508,116
C&C	-	57,486	-
DDoS	1,035,380	33,148,183	6,165,008
DoS	64,646	-	3,375,328
Injection	-	18	452,659
MITM	101,885	-	1,052
Password	-	-	1,718,568
Ransomware	-	-	72,805
Scanning	27,805	261,234,170	7,140,161
XSS	-	-	2,108,944
Benign	1,756,276	30,858,735	796,380

TABLE X
A PART OF TON-IOT DATA DISTRIBUTION

Category	**Training set**	**Testing set**
Backdoor	12000	8000
DDoS	12000	8000
DoS	12000	8000
Injection	12000	8000
Mitm	625	418
Password	12000	8000
Ransomware	12000	8000
Scanning	12000	8000
XSS	12000	8000
Benign	180000	120000

Among the most skewed features are: dst_ip_bytes, dst_pkts, $http\_response\_body\_len$ and $missed\_bytes$.

The tailedness of our data was estimated using a multivariate kurtosis function, as given by the following equation:

$$Kurtosis = \frac{\sum_{i=1}^{n}(x_i - \mu)^4}{n\sigma^4} \tag{4}$$

Kurtosis is a way to describe the shape of a probability distribution and measure extreme values with respect to the tails of the normal distribution. A high kurtosis indicates that more values of a feature distribution are close to the mean. In Fig. 4, the kurtosis values of the training and testing set are compared. The kurtosis values of the training and testing set only differ significantly for the same features as for skewness (Fig. 3).

V. FEATURE CORRELATIONS

The correlation of features is studied by means of two methods: without labels using the Pearson Correlation Coefficient (PCC), and with labels using the Information Gain Ratio (GR).

TABLE XI
TABLE OF SYMBOLS USED

Symbol	**Description**
x	Attribute or value in set
μ	Mean
σ	Standard deviation
$F(x)$	Empirical distribution function
$Sn(x)$	Cumulative distribution function
n	Number of records in an attribute
$cov(.)$	Covariance matrix
N	Random numbers for x
H_0, H_1	Null hypothesis and alternative hypothesis, respectively
p-value	Probability value used with H_0 or H_1
IG	Information gain
IV	Intrinsic value
p_i	Probability that a sample in a set belongs to a class

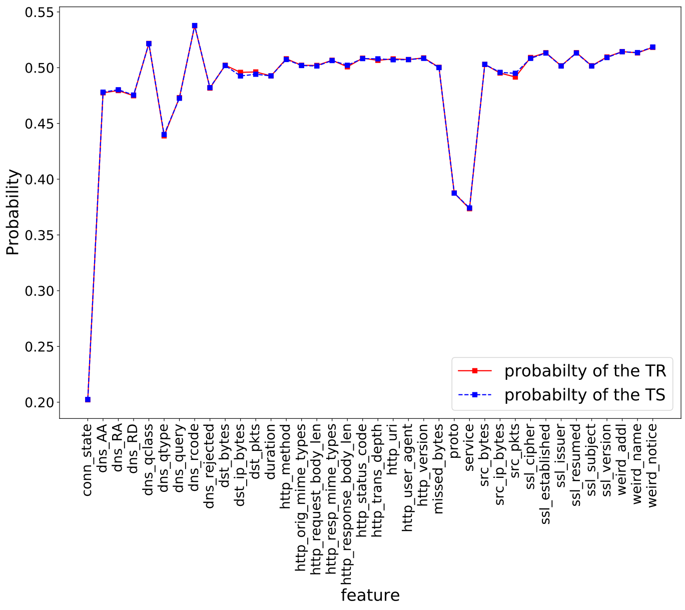

Fig. 2. The probability distribution of the features on the train and test sets.

PCC is computed for every attribute/feature in the training and testing set, as given by the following equation:

$$PCC(x_i, x_j) = \frac{cov(x_i, x_j)}{\sigma_{x_i} \cdot \sigma_{x_j}} \tag{5}$$

such that $cov(.)$ is the covariance matrix of any pair of attributes in X. In order to identify the strongest features, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3085194, IEEE Internet of Things Journal

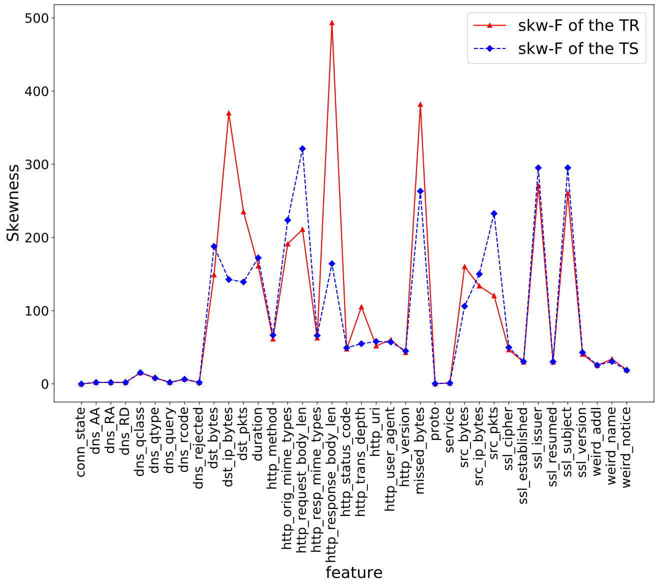

Fig. 3. The skewness of the features on the training and testing sets.

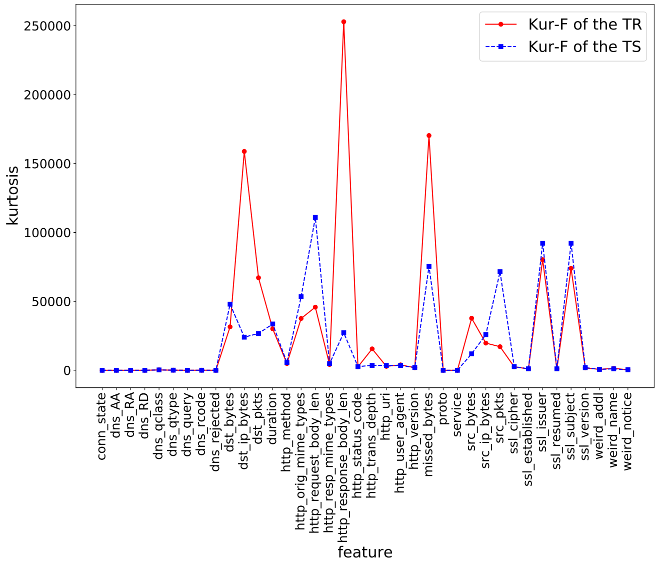

Fig. 4. The kurtosis of the features on the training and testing sets.

mean of PCC is calculated and plotted for each attribute x_i:

$$M_{PCC_i} = \sum_{j=1}^{N} \frac{PCC(x_i, x_j)}{N}. \tag{6}$$

As depicted in Fig 5, we see that all features have very similar (and low) correlation coefficients in both datasets.

A way to measure the correlation of each feature with the class label (i.e. label 0 designates benign traffic and label 1 is an malicious) is by determining the information gain. Information gain is a heuristic method to select attributes when building decision trees. The information gain ratio can be used to reduce the bias in the decision tree since attributes that can take on a large number of distinct values might dominate the learning process too much. It does so by considering the intrinsic information of a split.

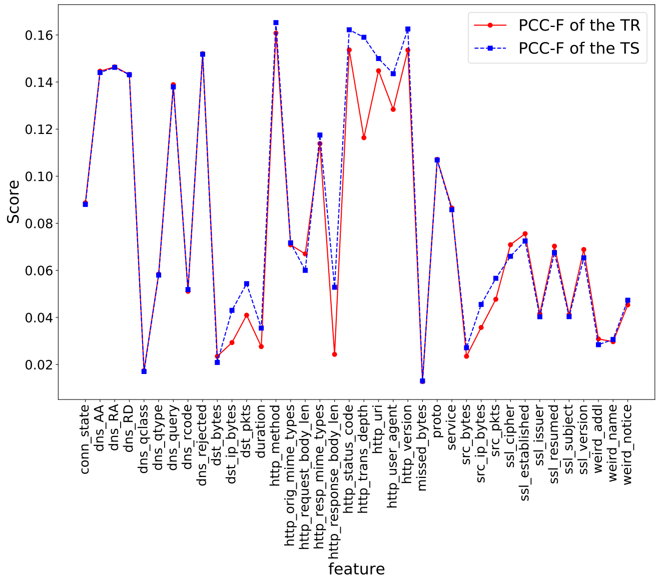

Fig. 5. The Pearson Correlation Coefficient (PCC) of the features on the training and testing sets.

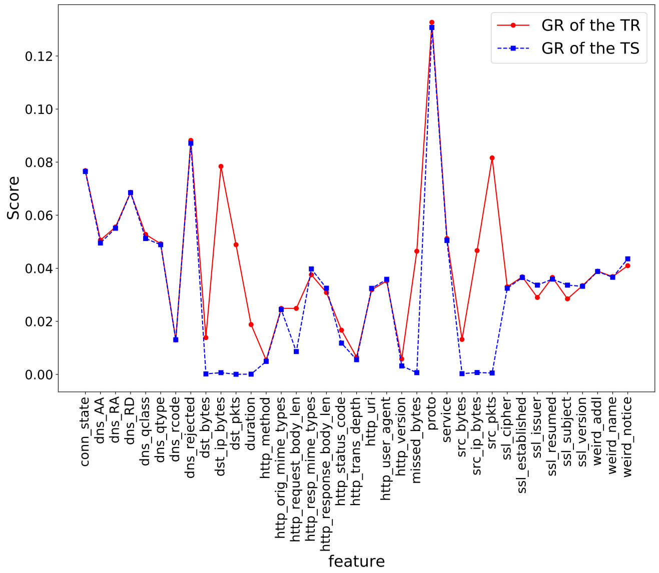

Fig. 6. The Information Gain Ratio (GR) of the features in the training and testing set.

If I is a randomly chosen instance in the set of training examples, $C = \{C_1, C_2, \ldots, C_w\}$ is the list of classes in the dataset, $X = \{x_1, x_2, \ldots, x_n\}$ is the set of attributes of I, $f_i = \{f_{i,1}, f_{i,2}, \ldots, f_{i,m}\}$ is the set of all possible values of attribute x_i and H specifies the entropy, then the information gain ratio GR is computed by taking the ratio between the information gain IG and the intrinsic value IV [31]:

$$GR(I, x_i) = \frac{IG(I, x_i)}{IV(I, x_i)} \tag{7}$$

$$= \frac{H(I) - H(I|x_i)}{IV(I)} \tag{8}$$

The entropy required to classify the observations is given by:

$$H(I) = -\sum_{i=1}^{w} p_i \log_2 p_i, \tag{9}$$

where p_i is the probability that a sample in the set I belongs to class C_i. If we let I_i be the subset of I containing data samples of class C_i. Then we have $p_i = I_i/I$. The conditional entropy is then calculated as the following equation:

$$H(I|x_i) = -\sum_{j=1}^{m} H(I)\frac{I_{1,j}+I_{2,j}+\ldots+I_{w,j}}{I} \tag{10}$$

where $I_{i,j}$ denotes the subset of I of samples from class C_i with feature x_i equal to $f_{i,j}$. Finally, the intrinsic value can be computed as:

$$IV(I) = -\sum_{j=1}^{m}\left|\frac{I_j}{I}\right|\log_2\left(\left|\frac{I_j}{I}\right|\right). \tag{11}$$

In our training and testing data there are a few features which assume a large amount of distinct values, making it very difficult to compute the gain ratio. These are: $duration$, src_bytes, dst_bytes, $missed\_bytes$, src_pkts, src_ip_bytes, dst_pkts, dst_ip_bytes, $http\_request\_body\_len$. To counter this problem, all values where sorted into classes by orders of magnitude: $[0,1)$, $[1,10)$, $[10,100)$, $\ldots$, $[10^{10},10^{11})$. Fig. 6 shows the GR of the features in the training and testing set. The GR for DNS queries has not been included in the graph because it was too computationally intensive to obtain: there are more than 10,000 different encoded domain names in both the training and testing set. Fig. 6 shows that most features are extremely similar in GR, again with the exception of the same set of features that also show discrepancies in skewness and kurtosis. The largest GR is attained by $dns\_rejected$ and $proto$.

VI. Classifier evaluation with ToN_IoT

A. Comparing between ToN_IoT and Aposemat IoT-23

To evaluate the effectiveness of the ToN_IoT dataset given its complexity and heterogeneity, we trained three different classifiers: Gradient Boosting Machine (GBM), Random Forest (RF) and a Multilayer Perceptron (MLP) which is a class of feedforward artificial Neural Network (NN), with ToN_IoT as well as Aposemat IoT-23. The classifiers were implemented with the default input parameters in R using the *h2o* package, and in Python using the *sklearn* package version 0.21.3 [30]. A summary of the results from this evaluation can be found in Table XII. Five metrics are shown. The F1 score is the harmonic mean of the precision and recall. The Area Under the Curve (AUC) measures the ability of a classifier to distinguish between classes. The Mean Square Error (MSE) is the average squared difference between the model and the actual data. Gini measures how often a randomly chosen element from the dataset would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset.

When comparing the results for both datasets, we can conclude that they both clearly provide high detection results, but the highest accuracy and F1-score is achieved with Aposemat IoT-23. This is not surprising if we take into account that Aposemat IoT-23 has significantly fewer features than ToN_IoT - 18 compared to 40: it is less heterogeneous and more specialized (particularly on botnet detection) than ToN_IoT. In other words, whatever can be detected with Aposemat IoT-23, is detected accurately, but there is also a lot that is not (and cannot be) detected. Given the significantly higher complexity and heterogeneity of ToN_IoT, we think that the obtained accuracy and F1-score are still rather good. Looking at the individual performance of the classifiers, we observe that Random Forest performs the best, with an AUC of 99.688% for ToN_IoT and 99.986% for Aposemat IoT-23. It is interesting to see that the Gradient Boosting Machine is outperformed by Random Forest. It is known that gradient boosting may not be a good choice if the data is very noisy, and our results indicate that this may be the case with both datasets. The supervised classifiers (GBM and RF) show somewhat better performance than the neural network MLP. Even better performance may be obtained by parameter optimization (we have acquired our results with default parameter settings using *sklearn* version 0.21.3). All-in-all, we can logically conclude that a higher heterogeneity will not result in a better performance.

B. Cross-training ToN_IoT and Aposemat IoT-23

A major weakness in how labeled datasets are typically evaluated in the common literature is given by the fact that samples for training and testing are always taken from the same set. This has practical reasons, as dataset creation and labeling is not standardized in any way. A trained classifier can therefore only be validated with data from the same set, and high performance results are often a logical consequence. Industry reality, however, is that (good) training sets are often not available. Instead, classifiers are trained on datasets such as the ones discussed in this article, leading to relatively low performance when the classifier is applied to actual data from the company's network. We therefore investigated how well the classifiers would perform when trained on ToN_IoT and tested with Aposemat IoT-23, and vice versa. The pipeline designed for this cross-training experiment is depicted in Figure 7.

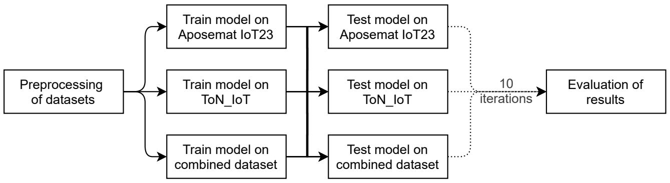
Preprocessing of datasets
Train model on Aposemat IoT23
Train model on ToN_IoT
Train model on combined dataset
Test model on Aposemat IoT23
Test model on ToN_IoT
Test model on combined dataset
10 iterations
Evaluation of results

Fig. 7. Analysis framework used throughout the cross-training comparison

Cross-training is still a relatively novel approach to the use of machine learning in network intrusion detection. An excellent overview of the state of the art has been published in 2018 by Guoquan Li et al [32]. The authors describe how cross-training by data fusion can be applied in three different ways or, as they say, on three different layers: the data layer,

TABLE XII
SUMMARY OF CLASSIFIER PERFORMANCES IN PERCENTAGES (%)
T = TON_IOT AND A = APOSEMAT IOT-23

Metrics	GBM		RF		NN (MLP)	
	T	A	T	A	T	A
Accuracy	94.643	99.452	98.075	99.986	97.842	99.420
F1	92.578	98.860	97.264	99.974	92.120	98.853
AUC	98.711	99.956	99.688	99.999	97.854	99.352
MSE	18.077	14.762	2.299	0.145	5.467	1.557
Gini	97.562	99.891	99.397	99.399	95.685	98.604

the feature layer, and the decision layer. In the data layer data from different datasets is processed and integrated into a new raw network dataset. Feature fusion happens on the feature layer. With feature fusing, only features are merged and often intelligently reduced to just a few but the most important features, which are then applied to both separate datasets. In the decision layer, the outcomes of the individual classifiers are fused and reduced to a single decision. An interesting outcome of the study in [32] is that in the field of network intrusion datasets, most cross-training by data fusion as published focus on the feature layer and the decision layer. The authors attribute this to most published work being based on open datasets, leading to data level fusion largely being omitted in the current literature.

In this section, we present an example of what can be achieved with cross-training by data fusion on the data level. We have chosen for this approach because of its analogy with industry practices, where companies often present unlabeled datasets from their network without baseline to research organizations for analysis. The only option these organizations then have is to train their model on an open source dataset, and applying it to the data at hand. With Aposemat IoT-23 and ToN IoT we are now mimicking that approach.

Given that Aposemat IoT-23 and ToN_IoT are inherently different, as would be the typical case in industry, we have trimmed the two datasets to their common features. In total, there are 13 features that both datasets have, namely: source port, destination port, protocol, service, duration, number of source bytes, number of destination bytes, connection state, number of missed bytes, number of source packets, number of destination packets and number destination IP bytes. The same classifiers listed in Table XII were trained on Aposemat IoT-23 and ToN_IoT and tested against the ToN_IoT and Aposemat IoT-23 test sets, respectively. We also concatenated both datasets to a combined set, and validated it against both test sets. These experiments are repeated 10 times, each with a newly created 60/40 split. The mean accuracy (μ) and standard deviation (σ), both in percentages, of the results are taken to remove positive/negative bias for the evaluation.

The results are shown in Table XIII. A first interesting observation is that classifiers trained on Aposemat IoT-23 and tested with ToN_IoT achieved a higher mean accuracy compared to the other way around. This can be explained by the fact that ToN_IoT includes more attack classes which cannot be recognized by models trained on Aposemat IoT-23. This is the same effect as discussed in the previous section regarding Table XII. Secondly, we see that classifiers trained on the combined dataset often achieve a slightly higher accuracy than those trained on the two datasets separately. This is explained by the fact that combining the datasets results in more diverse IoT traffic patterns. The increase in accuracy is not huge but significant (at least for GBM and RF, given the values of σ), and will still translate in malicious flows being detected which would otherwise go unnoticed.

This experiment has shown that training and testing on the same dataset mostly leads to very good performance outcomes, but training on one dataset and testing with the other provides far worse results. From this we conclude that the inclusion of as many as possible configurations is important. This is confirmed by the superb results obtained when concatenating both datasets. This analysis also shows that ToN_IoT has a higher heterogeneity than Aposemat IoT-23. The reasoning goes as follows. When using ToN_IoT as a testing set, a higher accuracy (true positives plus true negatives, divided by the total number of predictions) is obtained compared to using it as a training set (and testing it with Aposemat IoT-23). In the first case, the model is mostly trained on just botnets and scanning, and those will be correctly detected in the test set as true positives. The remainder is detected as true negatives. In the latter case, when training on ToN_IoT and testing on Aposemat IoT-23, botnets and scanning will again be correctly detected, but other attack types will sometimes be detected as false positives instead of true negatives.

VII. CONCLUSIONS

In this paper, we have analyzed the novel ToN_IoT dataset, with a particular focus on its heterogeneity. We conclude that ToN_IoT is pioneering current IoT network intrusion datasets, as it is the first to combine information from four heterogeneous sources - pcap files, Bro logs, sensor data and OS logs. Compared to other recently produced IoT network intrusion datasets, ToN_IoT incorporates a more diverse set of attack types and the highest number of different (types of) IoT devices. Besides prominent attacks such as DDoS and port scanning, more diversity is also found in other and more complex attacks, such as backdoors being exploited or cross-site scripting attacks. The statistical analysis of ToN_IoT

TABLE XIII
RESULTING MEAN ACCURACY AND STANDARD DEVIATION OF 10 ITERATIONS OF CROSS-TRAINING AND TESTING DATASETS

Model	GBM				RF				NN (MLP)			
	Test				Test				Test			
	Aposemat IoT-23		ToN_IoT		Aposemat IoT-23		ToN_IoT		Aposemat IoT-23		ToN_IoT	
	μ (%)	σ (%)	μ (%)	σ (%)	μ (%)	σ (%)	μ (%)	σ (%)	μ (%)	σ (%)	μ (%)	σ (%)
Train Aposemat IoT-23	99.891	0.002	63.409	0.460	99.922	0.002	60.548	1.104	**99.448**	0.009	43.875	3.818
Train ToN_IoT	30.711	0.279	**99.989**	0.003	30.756	0.065	**99.975**	0.005	20.879	3.818	95.143	0.765
Train combined	**99.897**	0.003	99.984	0.003	**99.931**	0.002	99.963	0.005	99.022	0.236	**95.587**	0.675

provided insights into the distribution of features between testing and training sets. This shows that, besides a few outliers including *amount of bytes*, a good balance can be found between the two sets, which is an important requirement for machine learning applications.

ToN_IoT displays a relatively high level of heterogeneity and, as such, relates more closely to real-world industry-grade IoT networks which typically encompass many different setups. We have made this visible by executing a novel way of comparing different datasets, namely by cross-training the classifiers. The experiment performed in Section VI-B has shown that training and testing on the same dataset predominantly leads to good performance, but training on one dataset and testing with the other results in poor performance. We thus conclude that the inclusion of many different configurations is paramount.

Cross-training by combining different IoT network intrusion datasets requires standardization of feature descriptions and attack classes, or at least their semantic interoperability. Such standards do currently not exist. We have seen that training classifiers on one dataset, and testing against a different dataset which has slightly different specifications for the same attack classes (e.g. if port scanning includes TCP or not), immediately leads to large discrepancies in performance results. This paper is a call for industry and academia to join forces and start the process of jointly defining features and attack classes to create truly heterogeneous and, therefore, effective IoT network intrusion detection datasets. Ideally, by using such heterogeneous datasets, effective (un)supervised detection methods for IoT networks can be developed and, subsequently, be deployed in operational environments without missing attacks such as happened in our cross-training experiment.

References

[1] H. N. Saha, A. Mandal, and A. Sinha, "Recent trends in the Internet of Things," in *IEEE 7th Annual Computing and Communication Workshop and Conference, CCWC 2017, Las Vegas, NV, USA, January 9-11, 2017*. IEEE, 2017, pp. 1–4. [Online]. Available: https://doi.org/10.1109/CCWC.2017.7868439

[2] Subrahmanyam Vasamsetti, "Future automobile an introduction of IoT," *International Journal of Trend in Research and Development*, 2017. [Online]. Available: http://rgdoi.net/10.13140/RG.2.2.15587.40484

[3] B. R. Stojkoska and K. V. Trivodaliev, "A review of Internet of Things for smart home: Challenges and solutions," *Journal of Cleaner Production*, vol. 140, pp. 1454–1464, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S095965261631589X

[4] M. Bacco, L. Boero, P. Cassarà, M. Colucci, A. Gotta, M. Marchese, and F. Patrone, "IoT Applications and Services in Space Information Networks," *IEEE Wirel. Commun.*, vol. 26, no. 2, pp. 31–37, 2019. [Online]. Available: https://doi.org/10.1109/MWC.2019.1800297

[5] H. Ahmadi, G. Arji, L. Shahmoradi, R. Safdari, M. Nilashi, and M. Alizadeh, "The application of internet of things in healthcare: a systematic literature review and classification," *Univers. Access Inf. Soc.*, vol. 18, no. 4, pp. 837–869, 2019. [Online]. Available: https://doi.org/10.1007/s10209-018-0618-4

[6] N. Durdević, A. Labus, Z. Bogdanović, and M. Despotović-Zrakić, "Internet of things in marketing and retail," *International Journal of Advances in Computer Science & Its Applications*, vol. 6, pp. 7–11, 01 2017.

[7] Unit 42, "2020 Unit 42 IoT Threat Report," Palo Alto Networks, Tech. Rep., 2020. [Online]. Available: https://unit42.paloaltonetworks.com/iot-threat-report-2020/

[8] H. Akram, D. Konstantas, and M. Mahyoub, "A comprehensive iot attacks survey based on a building-blocked reference model," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 3, 2018. [Online]. Available: http://thesai.org/Publications/ViewPaper?Volume=9&Issue=3&Code=ijacsa&SerialNo=49

[9] R. Mitchell and I. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 55:1–55:29, 2013. [Online]. Available: https://doi.org/10.1145/2542049

[10] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network Intrusion Detection for IoT Security Based on Learning Techniques," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019. [Online]. Available: https://doi.org/10.1109/COMST.2019.2896380

[11] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009, Ottawa, Canada, July 8-10, 2009*. IEEE, 2009, pp. 1–6. [Online]. Available: https://doi.org/10.1109/CISDA.2009.5356528

[12] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," *IEEE*, pp. 1–6, 2015. [Online]. Available: https://doi.org/10.1109/MilCIS.2015.7348942

[13] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy, ICISSP 2018, Funchal, Madeira - Portugal, January 22-24, 2018*, P. Mori, S. Furnell, and O. Camp, Eds. SciTePress, 2018, pp. 108–116. [Online]. Available: https://doi.org/10.5220/0006639801080116

[14] H. Kang, D. H. Ahn, G. M. Lee, J. D. Yoo, K. ho Park, and H. K. Kim, "IoT network intrusion dataset," 2019. [Online]. Available: http://dx.doi.org/10.21227/q70p-q449

[15] A. Parmisano, S. Garcia, and M. J. Erquiaga, "Stratosphere Laboratory. A labeled dataset with malicious and benign IoT network traffic," January 2020.

[16] 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, 2018. [Online]. Available: https://doi.org/10.1109/MPRV.2018.03367731

[17] S. García, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *Comput. Secur.*, vol. 45, pp. 100–123, 2014. [Online]. Available: https://doi.org/10.1016/j.cose.2014.05.011

[18] M. Pahl and F. Aubet, "All Eyes on You: Distributed Multi-Dimensional IoT Microservice Anomaly Detection," *IEEE Computer Society*, pp. 72–80, 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8584985

[19] 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, 2019. [Online]. Available: https://doi.org/10.1016/j.future.2019.05.041

[20] M. Pahl, "DS2OS traffic traces, IoT traffic traces gathered in a the DS2OS IoT environment," 2017, https://www.kaggle.com/francoisxa/ds2ostraffictraces.

[21] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "The BoT-IoT Dataset," 2018. [Online]. Available: https://dx.doi.org/10.21227/r7v2-x988

[22] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, "IoTPOT: Analysing the Rise of IoT Compromises," in *9th USENIX Workshop on Offensive Technologies, WOOT '15, Washington, DC, USA, August 10-11, 2015*, A. Francillon and T. Ptacek, Eds. USENIX Association, 2015. [Online]. Available: https://www.usenix.org/conference/woot15/workshop-program/presentation/pa

[23] J. Tetazoo, "Leaked Mirai SourceCode for Research/IoC," 2020, https://github.com/jgamblin/Mirai-Source-Code.

[24] "ToN_IoT dataset," March 2021, https://cloudstor.aarnet.edu.au/plus/s/ds5zW91vdgjEj9i.

[25] N. Moustafa, M. Keshk, E. Debie, and H. Janicke, "Federated TON_IoT Windows Datasets for Evaluating AI-based Security Applications," *arXiv preprint arXiv:2010.08522*, 2020.

[26] A. Alsaedi, N. Moustafa, Z. Tari, A. N. Mahmood, and A. Anwar, "TON_IoT Telemetry Dataset: A New Generation Dataset of IoT and IIoT for Data-Driven Intrusion Detection Systems," *IEEE Access*, vol. 8, pp. 165 130–165 150, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.3022862

[27] "Nessus Vulnerability Assessment," Tenable, 2020. [Online]. Available: https://www.tenable.com/products/nessus

[28] epsylon, "UFONet Denial of Service Toolkit," 2020. [Online]. Available: https://ufonet.03c8.net/

[29] epsylon, "XXSer Cross Site Scripter," 2020. [Online]. Available: https://xsser.03c8.net/

[30] T. M. Booij and I. Chiscop, "Statistical analysis ToN_IoT Datasets," 2021. [Online]. Available: https://dx.doi.org/10.21227/frw4-sk06

[31] T. M. Cover and J. A. Thomas, *Elements of information theory (2. ed.)*. Wiley, 2006. [Online]. Available: http://www.elementsofinformationtheory.com/

[32] G. Li, Z. Yan, Y. Fu, and H. Chen, "Data fusion for network intrusion detection: A review," *Secur. Commun. Networks*, vol. 2018, pp. 8 210 614:1–8 210 614:16, 2018. [Online]. Available: https://doi.org/10.1155/2018/8210614

Tim Booij received his Bachelor (2016) and Master (2019) degree in Computer Science at the faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology in the Netherlands. During his masters, he specialised towards data science and received an additional cybersecurity degree from the 4TU.Federation. He contributed to multiple publications on cybercrime investigations and Cyber Threat Intelligence topics. He joined TNO directly after his studies, and has since worked as a cybesecurity researcher at the Cyber Security & Robustness department. His research mainly focuses on anomaly detection, incorporating supervised and unsupervised methods to find malicious activities in various domains, and research towards cybercrime on topics such as phishing or dark web.

Irina Chiscop received her Bachelor degree (2016) in Mathematics at the University of Groningen, and her Master degree (2018) in Applied Mathematics at Delft University of Technology. She joined TNO in 2018 and has since worked as a Scientist in the Cyber Security & Robustness department. Her research interests include machine learning, network intrusion detection, IoT/OT security and quantum computing.

Erik Meeuwissen received M.S. and Ph.D. degrees in electrical engineering from Eindhoven University of Technology in the Netherlands in 1992 and 1998, respectively. From 1998 to 2008 he was a senior member of technical staff at Bell Labs Europe in Hilversum, the Netherlands. Since 2008 he is a senior consultant Cyber Security and Robustness at TNO, the Netherlands. His research interests include mathematical modelling and optimization in the context of communication networks and services. He currently focuses on security monitoring & detection and anomaly detection & its applications. He served as overall project leader of the EQUANET consortium on End-to-End Quality of Service in Next-Generation Networks as well as the SeQual consortium on Composite Service Optimization and Quality in Service Oriented Architectures. Dr. Meeuwissen is a member of the Institute of Electrical and Electronics Engineers (IEEE). He is (co-)author of over 35 refereed journal and conference papers, and (co-)inventor of 9 filed patent applications. He is a corecipient of the 2004 Bell Labs President's Gold Award.

Nour Moustafa (M'15–SM'19) received his PhD degree in the field of Cyber Security from UNSW Canberra in 2017. He obtained his Bachelor and Master degree in Computer Science in 2009 and 2014, respectively, from the Faculty of Computer and Information, Helwan University, Egypt. He is a Lecturer and leader of Intelligent Security at the School of Engineering & Information Technology, University of New South Wales Canberra (UNSW Canberra), Australia. He was a Post-doctoral Fellow at UNSW Canberra from June 2017 till December 2018. His areas of interest include Cyber Security and Artificial Intelligence, in particular, network security, IoT security, threat models, privacy preservation, digital forensics, intrusion detection, machine and deep learning and machine learning techniques. He has been awarded the 2020 prestigious Australian Spitfire Memorial Defence Fellowship award. He is also a Senior IEEE Member, ACM Distinguished Speaker, as well as CSCRC Fellow. He is a founding member of the IEEE TEMS Technical Committee on Blockchain and Distributed Ledger Technologies. He has served his academic community, as the guest associate editor of IEEE transactions journals, including IEEE Transactions on Industrial Informatics, IEEE IoT Journal, as well as the journals of Ad Hoc Networks, IEEE Access, Future Internet and Information Security Journal: A Global Perspective. He has also served over seven conferences in leadership roles, involving vice-chair, session chair, program committee member, and proceedings chair, such as IEEE TrustCom, Australasian Joint Conference on Artificial Intelligence, and National Cyber Summit

Frank den Hartog (Senior Member, IEEE) received the M.Sc. degree in applied physics from the Eindhoven University of Technology, Eindhoven, The Netherlands, in 1992, and the Ph.D. degree in physics from Leiden University, Leiden, The Netherlands, in 1998. From 1998 to 2003, he worked for the Dutch incumbent telecom operator KPN. He was a Senior Scientist with the research organization TNO, The Netherlands, until 2016, where he acquired and led various large collaborative research projects in the field of smart homes and Internet of Things (IoT). From 2012 to 2016, he was the Chair of the Technical Working Group of the worldwide Home Gateway Initiative (HGI) industry consortium. In 2018, he became an Associate Professor at the University of New South Wales (UNSW), Canberra, Australia, specializing in complex systems security. He is now the Director of Postgraduate Studies with the School of Engineering and Information Technology, UNSW. He has coauthored 72 peer-reviewed articles, 67 contributions to standardization, and seven international standards. He holds 16 patents in Europe and the US. Prof. Den Hartog is a member of ACM SIGCOMM, TelSoc, AISA, and the IEEE CCNC Steering Committee.