Federated TON_IoT Windows Datasets for Evaluating AI-based Security Applications

Nour Moustafa*, Marwa Keshk†, Essam Debie‡ and Helge Janicke§
School of Engineering and Information Technology, University of New South Wales, Canberra, Australia *†‡
Cyber Security Cooperative Research Centre, Edith Cowan University, Perth, Australia§
Email: *nour.moustafa@unsw.edu.au, †marwa.hassan@student.unsw.edu.au,
‡e.debie@unsw.edu.au, §helge.janicke@cybersecuritycrc.org.au

***Abstract*—Existing cyber security solutions have been basically developed using knowledge-based models that often cannot trigger new cyber-attack families. With the boom of Artificial Intelligence (AI), especially Deep Learning (DL) algorithms, those security solutions have been plugged-in with AI models to discover, trace, mitigate or respond to incidents of new security events. The algorithms demand a large number of heterogeneous data sources to train and validate new security systems. This paper presents the description of new datasets, the so-called ToN_IoT, which involve federated data sources collected from Telemetry datasets of IoT services, Operating system datasets of Windows and Linux, and datasets of Network traffic. The paper introduces the testbed and description of TON_IoT datasets for Windows operating systems. The testbed was implemented in three layers: edge, fog and cloud. The edge layer involves IoT and network devices, the fog layer contains virtual machines and gateways, and the cloud layer involves cloud services, such as data analytics, linked to the other two layers. These layers were dynamically managed using the platforms of software-Defined Network (SDN) and Network-Function Virtualization (NFV) using the VMware NSX and vCloud NFV platform. The Windows datasets were collected from audit traces of memories, processors, networks, processes and hard disks. The datasets would be used to evaluate various AI-based cyber security solutions, including intrusion detection, threat intelligence and hunting, privacy preservation and digital forensics. This is because the datasets have a wide range of recent normal and attack features and observations, as well as authentic ground truth events. The datasets can be publicly accessed from this link [1].**

***Index Terms*—Federated datasets, AI-based security applications, testbed, Windows operating systems, intrusion detection.**

I. Introduction

Through technological advancement in the ICT sector, the Internet of Things (IoT) is becoming an important aspect of society to offer automated services to users and organizations. As IoT systems are assigned ever more important tasks, often handling sensitive data or critical infrastructure, it is evident that any disruption caused by malicious actors will result in tremendous problems such as loss of businesses or human lives [2]. As a result, shielding these technologies through cyber security and ensuring their stability is imperative. Due to the reliance on the Internet, there are many security vulnerabilities in IoT networks and their operating systems, such as Windows and Linux, where attackers could exploit systems to gain access and steal important information from them [3]–[5]. As a result, tremendous cyber attacks are expected to target IoT systems through operating systems because the IoT systems are easy to use and have been designed to be continuously on-line, providing a reliable platform from which to launch complex hacking events.

The Operating System (OS) is the core of computer systems and their networks that manage hardware underpinning and provide low-level services to high-level programs. Although different types of OSs have been developed over the years, in this paper we focus on Windows OS, especially Windows 7 and Windows 10. The Windows OS is a family of proprietary operating systems, owned and maintained by Microsoft [6]. Windows OSs available in various computer systems include Windows 7, Windows 10, Windows Server, Windows NT, Windows IoT, and Windows Mobile. Windows is the world's most used OS [7], [8], and has often been the target of cyber-attacks like ransomware and malware [9]–[11] with significant consequences.

Investigating cyber threats targeting Windows OS and developing new Artificial Intelligence (AI)-based cyber security solutions, including intrusion detection, threat intelligence and hunting, privacy preservation and digital forensics, have a great interest to improve system security. Multiple cyber threats have encouraged the design and development of various cyber defensive mechanisms, with specialized techniques that focus on specific device subsystems [12], [13]. Intrusion Detection Systems (IDS) are one particular family of defensive mechanisms that have been the focus of much research in recent years. [14]. IDSs can be categorized into two types: 1) Host IDSs (HIDSs) are installed and monitor a host machine; and 2) Network IDSs are placed in strategic locations of a network, monitoring inbound and outbound traffic. Depending on their inner mechanism, IDS are characterized as either being anomaly-based, flagging unusual behavior, or signature-based, detecting known patterns of misuse [13].

In order to evaluate IDSs and AI-based security solutions, it is vital to use high-quality data that realistically represent current behavioral scenarios, including both attacks and le-

This work was funded by Australian Research Data Commons (*RG192500*) and UNSW Canberra (*PS51776*).

gitimate events. Existing datasets that can be used for the development of a HIDS face several issues. To begin with, research has been focused on Linux OS, with most of the available datasets being derived from Linux-based testbeds that are different from audit traces of Windows OSs. Furthermore, existing datasets were generated from testbeds that did not incorporate IoT scenarios, where multiple lightweight devices communicate with each other to generate diverse normal patterns from conventional network systems [15], [16]. This is a considerable weakness, as it is exceedingly common to have smart devices active in a network, whether it is public or private. In addition, most datasets were designed to be used for HIDS development, focus entirely on API calls and system calls [15], neglecting data derived from other subsystems, such as memory, processor, process and hard disk, which would cause that the defensive mechanisms often ignore complex attacks. Finally, most of the existing datasets suffer from low credibility because they do not have the ground truth of security events or the provided data analysis is poor [15], [17].

This paper addresses the issues above by introducing new Windows datasets that comprise new features extracted from the audit traces of memory, processor, process and hard disk in a new IoT network architecture. The testbed was deployed in three tiers, edge, fog and cloud. The edge tier includes IoT and network devices, the fog layer involves virtual machines and gateways, and the cloud tier comprises cloud services such as data analytics and visualization linked to the other tiers. These tiers were elastically managed using the technologies of software-Defined Network (SDN) and Network-Function Virtualization (NFV) using the VMware NSX [18] and vCloud NFV [19] platforms. While configuring and deploying the testbed, normal and attack events were executed to gather labeled data samples based on an authentic ground truth table for evaluating the performances of new cyber security applications.

The main contributions of this paper are as follows:

- A new IoT testbed architecture is proposed for concurrently collecting federated data from heterogeneous sources, along with recent normal and malicious scenarios.
- Software-Defined Network connectivity and Network Function Virtualization are proposed in the testbed to offer dynamic virtualization services in the proposed testbed.
- New Windows-derived datasets that incorporate a wide range of recent legitimate and adversarial events were generated from the activities of hard disks, memories, processes, processors and network-level, for the validation of new AI-based cyber security solutions such as HIDSs.

The structure of this paper is as follows. Section II describes the background and related work on IDSs and their datasets. Section III illustrates the testbed designed for generating the TON_IoT datasets. Hacking scenarios used in the datasets are explained in Section IV. Following that, the features and statistics of Windows 7 and 10 datasets are explained in Section V. Finally, Section VI presents concluding remarks of the work and future research directions.

II. Background and Related work

This section provides background information about IDSs and their previous studies. Several existing datasets are also explained for the purpose of constructing and evaluating IDSs in Windows environments.

A. Intrusion Detection Systems (IDSs)

IDSs can be classified into two main categories, network-based and host-based IDS [12], [20]. Network-based IDSs (NIDSs) are platforms placed in a network's strategic points, such as gateways, to monitor traffic sent to and from the internal network for any suspicious patterns. Host-based IDSs (HIDS) are software agents to track host computers, relying on data, such as system calls and logs, to detect unauthorized activities.

An IDS can also be categorized on the basis of the detection method used in three categories: signature-based detection, anomaly-based detection, and hybrid of the first two. A signature-based detection [21] utilizes pre-defined patterns, known as signatures, which can be used to identify attacks. The idea of using signatures to detect intrusions is that a database of patterns is maintained and regularly updated to match and detect attack events. A signature-based HIDS monitors a host's state by scanning various logs, memory dumps and network traffic generated or received by the host. They produce a high detection rate for known attacks and produce results at high speeds. However, signature-based solutions are hindered by even small changes to the known attack patterns (a common defensive technique employed by attackers). In addition, as they rely on known patterns, signature-based techniques are unable to detect unknown attack patterns, also known as zero-day exploits [22].

An Anomaly-based HIDS, on the other hand, establishes a profile of legitimate usage, then flagging any activities that deviate from the profile as an intrusion [22]. As the decision engine, they make use of the machine and deep learning models, which are trained on collected data and learn to detect legitimate behavior. Because anomaly-based HIDSs fit normal behavior, they are capable of detecting zero-day attacks. In addition, they are capable of detecting mutations in attack patterns, a process often employed by malware to circumvent signature-based IDSs [23].

Although relying on the machine learning and deep learning models render anomaly-based HIDS more versatile compared to signature-based ones, they are also more computationally intensive [22], [24]. Additionally, signature-based techniques could outperform anomaly-based models, when tasked with detecting known patterns. The higher false alarm rates produced by anomaly-based techniques would come as a result of the generalization that machine learning techniques rely on, to make predictions on unknown data points [25]. Building

effective cyber security solutions, for example, intrusion detection, privacy preservation, threat hunting and intelligence and digital forensics, require proper datasets that include a wide range of recent legitimate and attack events to train and validate the security solutions effectively.

B. Datasets used for validating defensive mechanisms

Acquiring a reliable dataset, which includes contemporary attack and normal scenarios, is an important first step towards designing, developing and validating effective AI-enabled defensive applications such as intrusion detection, privacy preservation, threat hunting and intelligence and digital forensics. Several datasets have been proposed in the literature which can be employed in IDS development, with particular attention given on operating system datasets [26], [27]. The most commonly used datasets are briefly discussed as follows:

- The DARPA 98 and KDD-99 dataset [17] – is the first dataset intended for IDS research. The entire dataset includes 4GB of raw pcap files and was derived from a testbed that was modeled by the MIT Lincoln Laboratory. A number of attacks targeted Unix machines located in the testbed's network, within a period of seven weeks. Although the dataset is labeled and widely used to date, the attack scenarios are out of date since they were generated in 1998, and do not reflect new normal and abnormal behaviors.
- The ADFA-LD dataset [15] – was produced by the cybersecurity centre at the University of New South Wales at ADFA. It is a Linux-based dataset generated from a testbed where an Ubuntu OS machine was targeted by a range of diverse attacks, such as password brute force, privilege escalation and Meterpeter-generated payloads. After scanning the Ubuntu machine, three datasets were produced raw system call data, the first for training on normal data, the second for validation on normal data and the last set comprised of attack data.
- The NGIDS-DS dataset [28] – was created by the cybersecurity centre at the University of New South Wales at ADFA. The dataset has a combination of network traces, and host-based data, produced by utilizing the IXIA perfect storm tool and scanning the cyber-range set-up for a period of four days, producing more than ninety million records of network and host data. In the testbed, two Ubuntu OS machines were incorporated, one of which collected network traffic and the other recorded process-related data with a particular focus on system calls.

Although the aforementioned datasets were derived from Linux OS systems, a few datasets derived from Windows machines have also been proposed, as explained below.

- The SSENet-2014 dataset [29] – was produced by extending the SSENet-2011 dataset. This dataset was generated by attacking a vulnerable Windows server, resulting in 28 attributes in total, with 9 basics, 9 network traffic and 10 host attributes. Although the statistics of the dataset were not provided, the dataset's attributes match those of KDD-99.
- The AWSCTD dataset [30] – was designed as an alternative to ADFA-WD in order to provide a current Windows-based dataset. To generate the dataset, Windows 7 guest machines were infected with a total of 10276 malware samples, producing 112.56 million audit traces.
- The ADFA-WD dataset [31] – was created by the cybersecurity centre at the University of New South Wales, in an attempt to provide a reliable windows-based dataset for IDS. In the data-generation phase, a Windows XP machine was targeted by several attacks, including a zero-day attack and system calls in the form of a dynamic link library and called function addresses were recorded.

While the research studies focused on developing Linux-based datasets, the limited number of Windows datasets demonstrates the need for more research and the creation of a new dataset that incorporates realistic networks involving multiple Windows operating systems, IoT and Industrial IoT (IIoT) services. More importantly, the existing datasets focus entirely on process system calls, ignoring other sources of traces like the memory, hard drive or the processor state. Furthermore, there is a lack of IoT-derived data representation in the existing datasets. This paper seeks to address these shortcomings by proposing a new Windows dataset from a new IoT testbed network designed at the IoT lab of the University of New South Wales, Canberra.

III. Proposed Testbed for generating federated Windows TON_IoT datasets

The proposed testbed architecture of the ToN_IoT datasets for gathering audit traces of Windows operating systems is shown in Figure 1. The testbed was designed based on interacting network and IoT systems with the three layers of edge, fog and cloud to mimic the realistic implementation of recent real-world IoT networks. The dynamics of the three layers, involving physical and simulated systems, were flexibly managed by the technologies of SDN and NVF. The NSX-VMware data center platform [18] was used to provide an SDN solution for the proposed testbed of the TON_IoT datasets. This technology permits the creation of overlay networks with the same capabilities of physical networks.

The VMware NSX platform was deployed with the VMware NFV hypervisor to allow the creation and management of various virtual machines that concurrently operate to offer the IoT and network services. In VMware NSX, the vCloud NFV platform was employed to provide a modular design with abstractions that enable multi-domain, hybrid physical and VM deployments [19]. The NSX vCloud NFV platform enables the design of a dynamic testbed IoT network of the ToN_IoT with creating and controlling several Virtual Machines (VMs) for hacking and normal operations, allowing the communications between the edge, fog and cloud layers.

The components of the testbed are explained for the three layers as follows:

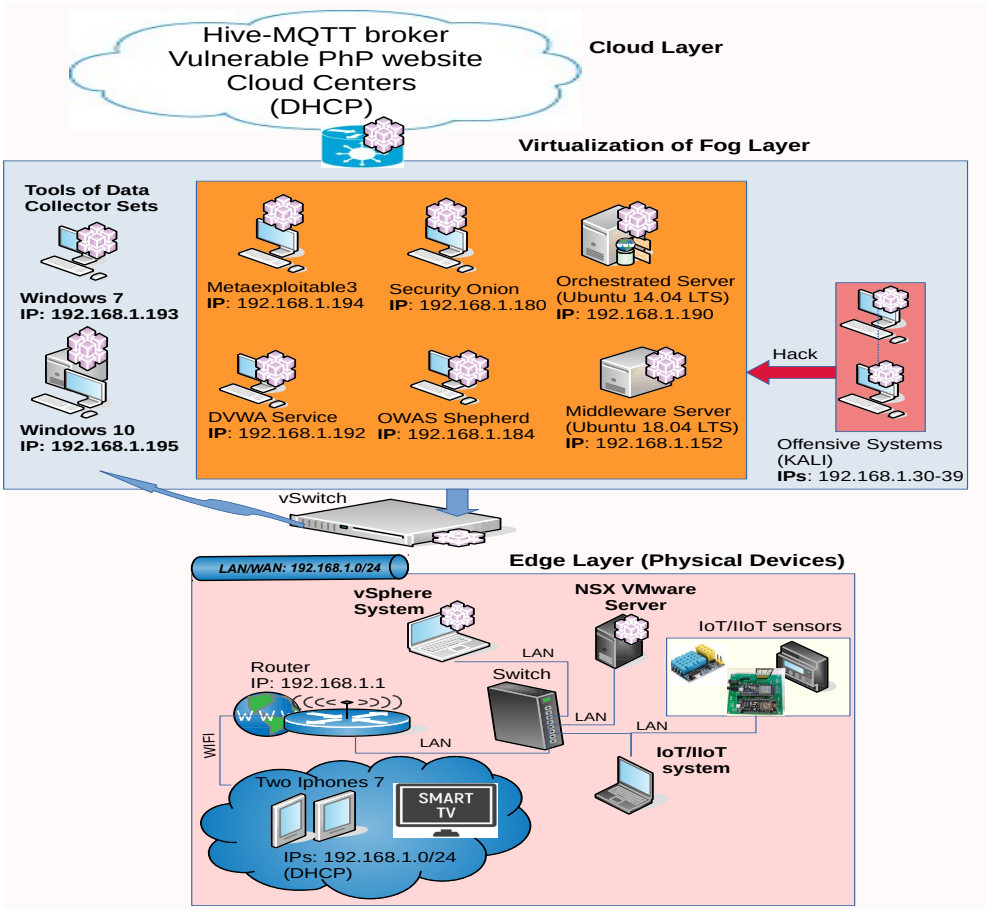

Fig. 1. Configured testbed of TON_IoT datasets for collecting Window data

1) **Edge layer** - involves the physical devices and their operating systems utilized as the infrastructure of configuring the virtualization technology and cloud services at the layers of fog and cloud, respectively. It includes multiple IoT/IIoT devices, such as Modbus, light bulb sensors, smartphones, smart TVs, as well as host systems, such as workstations and servers, used to connect IoT/IIoT devices, hypervisors and physical gateways (i.e., routers and switches) to the Internet. The hypervisor technology of NSX-VMware was installed on a host server at the edge layer to manage the Virtual Machines (VMs) created at the fog layer.
2) **Fog layer**- includes the virtualization technology that controls the VMs and their services using the NSX-VMware and vCloud platform to offers the framework of executing SDN and NFV in the proposed testbed. The NSX vCloud NFV platform enables the design of a dynamic testbed IoT/IIoT network of the ToN_IoT with creating and controlling several VMs for hacking and normal operations, allowing the communications between the edge, fog and cloud layers. This layer includes the nodes of virtual machines configured to generate the datasets, as explained in the following:

- ***Orchestrated server*** - is one of the main virtualized servers configured in the testbed using the Ubuntu 14.04 LTS with the IP address (*192.168.1.190*). This server offered many orchestrated services, such as FTP, Kerberos, HTTPS, and DNS to simulate real production networks and generate more simulated network traffic using the Ostinato Traffic Generator [32] that transmits traffic to other VMs in the testbed.
- ***Middleware server***- is the IoT/IIoT virtualized server deployed in the testbed using the Ubuntu 18.04 with the IP address (*192.168.1.152*). This server included the scripts that run IoT/IIoT services through public and local MQTT gateways utilized in the testbed and linked with the cloud layer to subscribe and publish the telemetry data of IoT/IIoT sensors.
- ***Client Systems*** - include a Windows 7 VM (IP address: *192.168.1.193*), Windows 10 VM (192.168.1.195), DVWA web service (*192.168.1.192*), OWASP security Sphered VM (192.168.1.184), Metaspoitable 3 (*192.168. 1.194*). The two windows were used as the remote web interface of the node-red IP (*192.168.1.152*)

and their network traffic and audit traces were logged. The Damn Vulnerable Web App (DVWA) [33] was utilized to make security vulnerabilities through web applications hacked using the virtualized offensive systems. The OWASP security Sphered VM [34] is an open-source platform that has many security vulnerabilities against mobile and web applications exploited using the offensive systems. In addition, the Metasploitable3 VM [35] was deployed in the testbed to increase vulnerable fog nodes and hack them using various attacking techniques by the offensive systems.

- ***Offensive systems*** - include the kali Linx VMs and scripts of hacking scenarios that exploit vulnerable systems in the testbed network. Ten static IP addresses (i.e.,*192.168.1.30-39*) were employed in the testbed to launch attacking scenarios and breach vulnerable systems either IoT/IIoT services (client and public MQTT brokers and node-red IP), operating systems (i.e., Windows 7 and 10, and Ubuntu 14.04 LTS and 18.04 LTS), and network systems (i.e., IP addresses and open protocols of the VMs).
- ***Data Logger Systems*** - log audit traces of Windows 7 and 10 operating systems included in the testbed. The Data Collector Set tool arranges data collection points, such as performance counters and event trace data, into a single collection [36]. Data Collector Sets enable us to schedule data collection so that the data can be anaysled and generated in CSV files, as in our datasets. While launching normal and hacking scenarios, the tools of Data Collector Sets in both Windows VMs were automatically configured to log data features of memories, networks, hard disks, processors and processes that happened. Data that were collected for performance counters by the tools of Data Collector Sets configured on Windows 7 and Windows 10 were stored in log files and can be opened using the Windows Performance Monitor tool (*.blg format*) [37]. The data features from generated for the Windows TON_IoT dataset are discussed below.

3) **Cloud layer**- contains the cloud services configured online in the testbed, as shown in Figure 1. The fog and edge services connected with the public HIVE MQTT dashboard [38], a public PHP vulnerable website [39], cloud virtualization, and cloud data analytics services (e.g., Microsoft Azure or AWS). The public HIV MQTT dashboard enabled us to publish and subscribe to the telemetry data of IoT/IIoT services via the configuration of the node-red tool. The public PHP vulnerable website used to launch injection hacking events against websites. The other cloud services were configured either in Microsoft Azure or AWS to transmit sensory data to the cloud and visualize their patterns.

IV. Hacking Techniques launched in Testbed

Hacking scenarios were utilized to launch nine attack categories against vulnerable elements of IoT/IIoT applications, operating systems and network systems. The scripts and some links of the attacking categories have been published in [1]. The nine attack families employed in the datasets are explained as follows:

1) **Scanning attack** - We used the Nessus and Nmap tools from the offensive systems with IP addresses (*192.168.1.20-38*) against the target subnet (*192.168.1.0/24*) and all other public vulnerable systems such as the Public MQTT broker and vulnerable PHP website. For example, nmap *192.168.1.40-254*, and the scans of the Nessus tool for the same range of IP addresses.
2) **Denial of Service (DoS) attack** - We utilized DoS attack scenarios on the offensive systems with IP addresses (*192.168.1.30,31,39*) to hack vulnerable elements in the IoT testbed network. We created Python scripts using the Scapy package to launch the DoS attacks.
3) **Distributed Denial of Service (DDoS) attack** - We used DDoS attacks in the offensive systems wit IP addresses (*192.168.1.30,31,34,35,36,37,38*) to breach several weaknesses in the IoT testbed network. We developed Python scripts using the Scapy package to launch the DoS attacks. Further, automated bash scripts were developed to launch DDoS against vulnerable nodes of the testbed using the ufonet toolkit.
4) **Ransomware attack** - We used the Kali Linux with IP addresses (*192.168.1.33, 37*) to execute this malware against windows operating systems and their webpages of monitoring IoT services included in the testbed network. This attack executed using the Metasploit framework that hacks the SMB vulnerability of the systems, named eternalblue.
5) **Backdoor attack** - We used the offensive systems with IP addresses (*192.168.1.33,37*) to keep the hacking persistence using the Metasploit framework by executing a bash script of the command “run persistence -h”.
6) **Injection attack** - We used various injection scenarios from the offensive systems with IP addresses (*192.168.1.30, 31, 33, 35*) to inject data inputs against web applications of DVWA and Security Shepherd VMs and webpages of IoT services through other VMs, including SQL injection, client-side injection, broken authentication and data management, and unintended data leakage.
7) **Cross-site Scripting (XSS) attack** - We employed the offensive systems with IP addresses (*192.168.1.32,35,36,39*) to illegally inject web applications of DVWA and Security Shepherd VMs and webpages of IoT services through other VMs. In these systems, we created malicious bash scripts of python codes to hack the web applications of the testbed

network using the Cross-Site Scripter toolkit (named XSSer).

8) **Password attack** - We used the offensive systems with IP addresses (*192.168.1.30, 31, 32, 35, 38*). In these systems, the hydra [40] and cewl toolkits were configured using automated bash scripts to concurrently launch password hacking scenarios against vulnerable nodes in the testbed.
9) **Man-In-The-Middle (MITM) attack** - We utilized the offensive systems with IP addresses (*192.168.1.31,34*) to launch various MITM scenarios in the testbed network. In the systems, we employed the Ettercap tool to execute ARP spoofing, ICMP redirection, port stealing and DHCP spoofing.

V. Experimental Results and Discussions

A. Feature Generation and Labelling

The Windows datasets were generated using the virtual machines running Windows 7 and Windows 10 and incorporated the collections of data from multiple sources, including memory, process, processor and hard drive of the systems. In order to generate the Windows datasets, the collectors of the Performance Monitor Tool [37] were executed on each machine. The initial raw version of the datasets was collected in a *blg* format, and through the Performance Monitor tool, the disk, process, processor, memory activities were extracted and saved in a *CSV* format. As explained in [1], the Windows 7 dataset generated 133 features and other two attributes of the class label, either normal or attack, and attack types of the nine attacked used in the testbed. The Windows 10 dataset generated 125 features and the other two attributes of the class label and attack types.

After generating the data features of Windows 7 and Windows 10, we added the two attributes for labeling each record either normal or attack type. The labeling process was used by the ground truth CSV files that contain the attack events that took place while running the testbed. The timestamp '*ts*' attribute was matched against each record in the CSV files, where if the '*ts*' of the ground truth table equals the '*ts*' of the data records in the CSV files, the records were labeled as attacks; otherwise, they were labeled as normal. The authentic labeling process of the datasets proves the fidelity of the correct security events that occurred during the implementation of the testbed and its authenticity for evaluating cyber security solutions based on machine learning algorithms.

B. Statistics of TON_IoT Windows datasets

There is a huge number of normal and attack types collected in the Windows 7 and Windows 10 datasets, as listed in Table I. It can be seen that the windows 7 dataset includes 28367 records while the Windows 10 dataset involves 35975 records of both normal and attack observations. These datasets include the CSV files of training-testing sets for training and testing machine learning models, as published in [1].

TABLE I
Number of normal records and each type of attach collected in the Windows 7 and 10 datasets

Type of events	Windows 7	Windows 10
Normal	22387	24871
Backdoor	1779	-
DDoS	2134	4608
Ransomware	82	-
Injection	998	612
XSS	4	1268
Password	757	3628
Scanning	226	447
DoS	-	525
MITM	-	15

TABLE II
10 most correlated features (their data types are numeric) in the Windows 7 dataset.

Feature name	Feature Description
Process_Total_IO Other_Bytes_sec	The process issues bytes to I/O operations that do not involve data such as control operations.
Network_I.Intel_R Pro_1000MT_Bytes Received_sec	The rate at which bytes are received over each network adapter, including framing characters.
Process_Total_IO Other Operations_sec	The rate at which the process issues I/O operations that are neither read nor write operations (for example, a control function).
Process_Total_IO Data Bytes_sec	The rate at which the process is reading and writing bytes in I/O operations.
Process_Total_IO Read Bytes_sec	The rate at which the process is reading bytes from I/O operations.
Network_I.Intel.R._ Pro_1000MT_Bytes Received_sec	The rate at which bytes are received over each network adapter.
Process_Pool_Paged_ Bytes	The size, in bytes, of the paged pool, an area of the system's virtual memory that is used for objects that can be written to disk when they are not being used.
Network_I.Intel.R_Pro_ 1000MT.Bytes_Sent_ sec	The rate at which bytes are sent over each network adapter, including framing characters.
Network_I.Intel.R_Pro_ 1000MT.Packets Received_sec	The rate at which packets are received on the network interface.
Process.Total_IO Data Operations_sec	The rate at which the process is issuing read and write I/O operations.

The training-testing set of Windows 7 involves 10,000 normal records and 5980 attack records, whilst the training-testing set of Windows 10 contains 10,000 normal records and 11104 attack records, as demonstrated in Table I. It is observed that all the attack types included in the entire sets are used in Table I. The training-testing sets were carefully analyzed to include different attack types and normal behaviors included in the entire sets. **Researchers can then divide these sets for two subsets, such as 70% for training and 30% for testing or k-fold cross-validation models, for testing the performances of AI-enabled cyber security models.**

C. Correlation Analysis of Features

The correlation analysis has a great impact to demonstrate the strength of features and their utility to define security events using machine learning models. In order to estimate the correlation coefficient between the features without the label

TABLE III
10 MOST CORRELATED FEATURES (THEIR DATA TYPES ARE NUMERIC) IN THE WINDOWS 10 DATASET.

Feature name	Feature Description
Network_I.Intel.R_ 82574L_GNC.Current Bandwidth	The current bandwidth of the network interface in bits per second (BPS).
Network_I.Intel.R_ 82574L_GNC.Packets Sent Unicast.sec	The rate at which packets are requested to be transmitted to subnet-unicast addresses by higher-level protocols.
Memory.Pool.Paged Bytes	The size, in bytes, of the portion of the paged pool that is currently resident and active in physical memory.
LogicalDisk._Total Disk Read Bytes.sec	The rate at which bytes are transferred from the disk during reading operations.
Memory.Page Reads.sec	The rate at which the disk was read to resolve hard page faults.
Network_I.Intel.R_ 82574L_GNC.Packets Sent.sec	The rate at which packets are sent on the network interface.
Memory.Modified. Page List Bytes	The amount of physical memory, in bytes, that is assigned to the modified page list.
Process_IO Data Operations_sec	The rate at which the process is issuing read and write I/O operations.
LogicalDisk_Total. Avg.Disk.Bytes Transfer	The average number of bytes transferred to or from the disk during write or read operations.
Processor_pct_ Processor_Time	The amount of elapsed time that the processor spends executing a non-Idle thread.

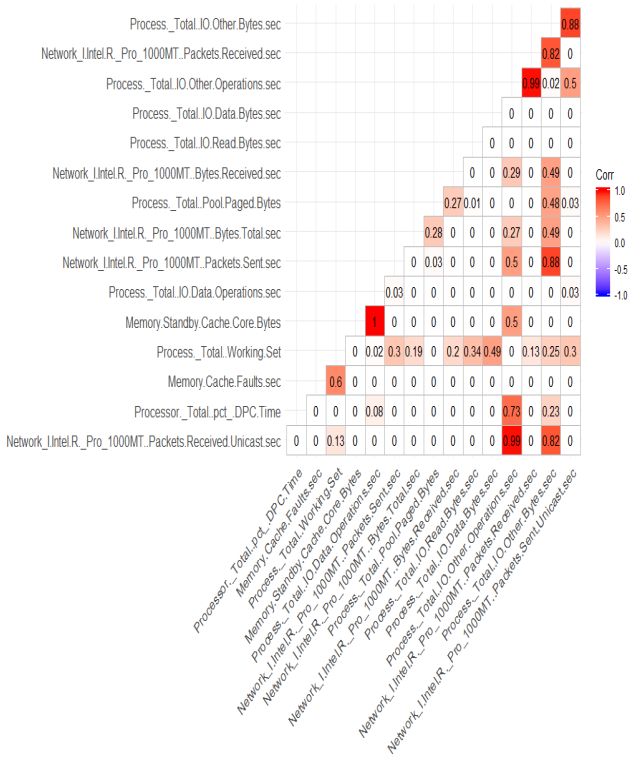

Fig. 2. Correlation matrix of most important features in the Windows 7 dataset

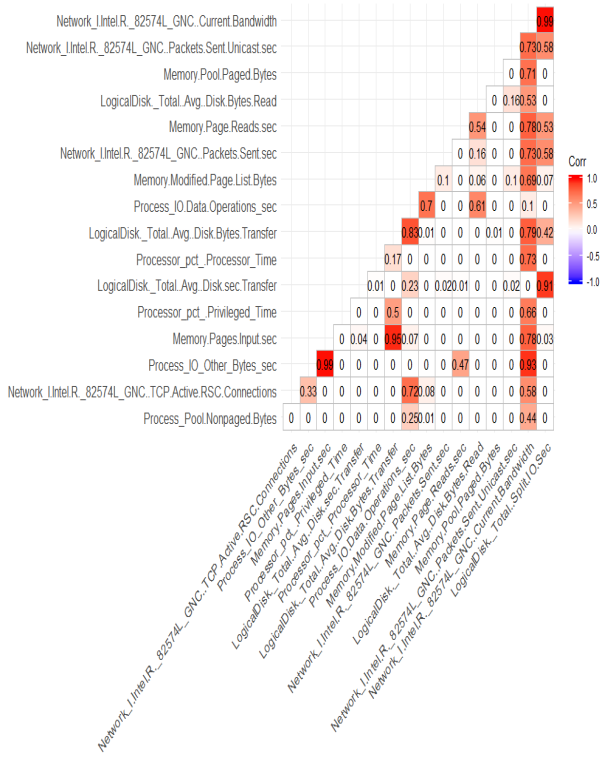

Fig. 3. Correlation matrix of most important features in the Windows 10 dataset

attributes on the Windows 7 and 10 datasets, we developed a correlation coefficient function [26] in the R programming language to rank the features' strengths into a range of [-1, 1].The sign of the correlation coefficient refers to the direction of the relationship, while the magnitude of the correlation (i.e, how close it is to -1 or +1) indicates the strength of the relations between the features [26].

The correlation matrix was adapted to select the most correlated features with higher than or equal a cut-off value of 0.85%. The most 10 correlated features in both datasets are described in Tables II and III. The description of the rest features can be found in [1]. The most correlated features of the windows 7 and 10 datasets are shown in Figures 2 and 3, respectively. The most correlated features would be used for training and validating machine/deep learning algorithms to evaluate their efficiency in classifying attack families included in the datasets.

VI. CONCLUSION

This paper has introduced the description and preliminary results of the Windows TON_IoT datasets created at the IoT lab of UNSW Canberra. In order to create the federated datasets, a new IoT testbed was designed that included a wide variety of IoT services deployed at the edge layer, virtual machines of operating systems configured at the fog layer, as well as cloud services constructed at the cloud layer. The dynamic interaction between the three layers was deployed using the VMware NSX and vCloud NFV platforms

to provide SDN and NVF services. Recent normal and nine attack categories were executed in the datasets in order to generate authentic data sources for assessing the reliability of new AI-based cyber security applications. Further, the features of Windows 7 and Windows 10 data were collected from the audit traces of memories, processors, processes and hard disks to ensure the identification of new attack patterns that could stealthy exploit windows operating systems. A large number of data samples was collected for windows 7 and windows 10 datasets. The collected datasets show a wide variety of normal and attack events, revealing the fidelity of the datasets for assessing new AI-based cyber security applications, including intrusion detection, privacy preservation, digital forensics, as well as threat intelligence and hunting, in which we will explore in the future.

REFERENCES

[1] "Ton_iot dataset," *https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-ton-iot-Datasets/*, January 2020.

[2] N. Moustafa, "A systemic iot-fog-cloud architecture for big-data analytics and cyber security systems: A review of fog computing," *arXiv preprint arXiv:1906.01055*, 2019.

[3] G. Ho, D. Leung, P. Mishra, A. Hosseini, D. Song, and D. Wagner, "Smart locks: Lessons for securing commodity internet of things devices," in *Proceedings of the 11th ACM on Asia conference on computer and communications security*. ACM, 2016, pp. 461–472.

[4] Y. Seralathan, T. T. Oh, S. Jadhav, J. Myers, J. P. Jeong, Y. H. Kim, and J. N. Kim, "Iot security vulnerability: A case study of a web camera," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*. IEEE, 2018, pp. 172–177.

[5] N. Koroniotis, N. Moustafa, and E. Sitnikova, "Forensics and deep learning mechanisms for botnets in internet of things: A survey of challenges and solutions," *IEEE Access*, vol. 7, pp. 61 764–61 785, 2019.

[6] P. Yosifovich, D. A. Solomon, and A. Ionescu, *Windows Internals, Part 1: System architecture, processes, threads, memory management, and more*. Microsoft Press, 2017.

[7] A. Majeed and S. Saleem, "Forensic analysis of social media apps in windows 10," *NUST Journal of Engineering Sciences*, vol. 10, no. 1, pp. 37–45, 2017.

[8] A. M. Aladdin, Y. N. Bakir, S. I. Saeed *et al.*, "The effects to trend the suitable os platform," *JOURNAL OF ADVANCES IN NATURAL SCIENCES*, vol. 5, pp. 342–351, 2018.

[9] A. Akkas, C. N. Chachamis, and L. Fetahu, "Malware analysis of wanacry ransomware," 2017.

[10] D. Kim, B. J. Kwon, and T. Dumitraş, "Certified malware: Measuring breaches of trust in the windows code-signing pki," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 1435–1448.

[11] A. Zimba, Z. Wang, and H. Chen, "Multi-stage crypto ransomware attacks: A new emerging cyber threat to critical infrastructure and industrial control systems," *Ict Express*, vol. 4, no. 1, pp. 14–18, 2018.

[12] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[13] N. Moustafa, G. Creech, and J. Slay, "Big data analytics for intrusion detection system: Statistical decision-making using finite dirichlet mixture models," in *Data analytics and decision support for cybersecurity*. Springer, 2017, pp. 127–156.

[14] N. Moustafa, G. Creech, E. Sitnikova, and M. Keshk, "Collaborative anomaly detection framework for handling big data of cloud computing," in *2017 Military Communications and Information Systems Conference (MilCIS)*. IEEE, 2017, pp. 1–6.

[15] G. Creech and J. Hu, "Generation of a new ids test dataset: Time to retire the kdd collection," in *2013 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2013, pp. 4487–4492.

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

[17] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham *et al.*, "Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation," in *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, vol. 2. IEEE, 2000, pp. 12–26.

[18] "Vmware sdn," *https://lenovopress.com/lp0661.pdf*, January 2020.

[19] "Vmware nfv," *https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/products/nfv/vmware-vcloud-nfv-vcloud-director-edition-datasheet.pdf*, January 2020.

[20] H. Kozushko, "Intrusion detection: Host-based and network-based intrusion detection systems," *Independent study*, 2003.

[21] S. Freeman, A. Bivens, J. Branch, and B. Szymanski, "Host-based intrusion detection using user signatures," in *Proceedings of the Research Conference. RPI, Troy, NY*, 2002, pp. 2005–2014.

[22] S. Jose, D. Malathi, B. Reddy, and D. Jayaseeli, "A survey on anomaly based host intrusion detection system," in *Journal of Physics: Conference Series*, vol. 1000, no. 1. IOP Publishing, 2018, p. 012049.

[23] N. Moustafa, K.-K. R. Choo, I. Radwan, and S. Camtepe, "Outlier dirichlet mixture mechanism: Adversarial statistical learning for anomaly detection in the fog," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 8, pp. 1975–1987, 2019.

[24] M. Bijone, "A survey on secure network: intrusion detection & prevention approaches," *American Journal of Information Systems*, vol. 4, no. 3, pp. 69–88, 2016.

[25] N. Moustafa, G. Creech, and J. Slay, "Anomaly detection system using beta mixture models and outlier detection," in *Progress in Computing, Analytics and Networking*. Springer, 2018, pp. 125–135.

[26] 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 Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

[27] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 military communications and information systems conference (MilCIS)*. IEEE, 2015, pp. 1–6.

[28] W. Haider, J. Hu, J. Slay, B. P. Turnbull, and Y. Xie, "Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling," *Journal of Network and Computer Applications*, vol. 87, pp. 185–192, 2017.

[29] S. Bhattacharya and S. Selvakumar, "Ssenet-2014 dataset: A dataset for detection of multiconnection attacks," in *2014 3rd International Conference on Eco-friendly Computing and Communication Systems*. IEEE, 2014, pp. 121–126.

[30] D. Čeponis and N. Goranin, "Towards a robust method of dataset generation of malicious activity for anomaly-based hids training and presentation of awsctd dataset," *Baltic Journal of Modern Computing*, vol. 6, no. 3, pp. 217–234, 2018.

[31] W. Haider, G. Creech, Y. Xie, and J. Hu, "Windows based data sets for evaluation of robustness of host based intrusion detection systems (ids) to zero-day and stealth attacks," *Future Internet*, vol. 8, no. 3, p. 29, 2016.

[32] "Ostinato traffic generator," *https://ostinato.org/*, January 2020.

[33] "Dvwa web service," *http://www.dvwa.co.uk/*, January 2020.

[34] "Owasp security shepherd," *https://owasp.org/www-project-security-shepherd/*, January 2020.

[35] "Metasploitable3 vm," *https://github.com/rapid7/metasploitable3*, January 2020.

[36] "Data collector sets of windows," *https://docs.microsoft.com/en-us/dynamics-nav/working-with-data-collector-sets*, January 2020.

[37] "Windows monitor tool," *https://help.tableau.com/current/server/en-us/perf_collect_perfmon.html*, January 2020.

[38] "Public hive mqtt broker," *https://www.hivemq.com/public-mqtt-broker/*, January 2020.

[39] "Public php vulnerable website," *http://testphp.vulnweb.com/*, January 2020.

[40] "Hydra tool," *https://tools.kali.org/password-attacks/hydra*, January 2020.