A Novel SDN Dataset for Intrusion Detection in IoT Networks

Alper Kaan Sarica
Department of Computer Engineering
Middle East Technical University
Ankara, Turkey
kaan.sarica@metu.edu.tr

Pelin Angin
Department of Computer Engineering
Middle East Technical University
Ankara, Turkey
pangin@ceng.metu.edu.tr

***Abstract*—The number of Internet of Things (IoT) devices and the use cases they aim to support have increased sharply in the past decade with the rapid developments in wireless networking infrastructures. Despite many advantages, the widespread use of IoT has also created a large attack surface frequently exploited by cyber criminals, requiring real-time, automated detection and mitigation of various attacks in the high-volume network traffic generated. Software-defined networking (SDN) and machine learning (ML) based intrusion detection are effective tools for providing quick response to various attacks in IoT networks, however the study of ML-based intrusion detection so far has been limited to performance studies on datasets that were created a long while ago and are not specific to SDN-based environments. In this paper we introduce a novel dataset for intrusion detection in IoT networks. The dataset comprises two parts modeling static and dynamic IoT networks and consists of 27.9 million and 30.2 million data records respectively, which contain cyber attacks of various types in addition to benign traffic. The dataset will be an important resource for intrusion detection research in SDN-managed IoT, which will be increasingly prevalent in the future networks of ubiquitous connectivity.**

***Index Terms*—Software-defined networking; machine learning; dataset; security**

I. Introduction

The number of connected devices and IoT use cases have been increasing rapidly with the advances in wireless networks, data analytics techniques and cloud platforms. IoT use cases include connected cars, smart grids, video surveillance and security systems, smart homes, smart cities and smart healthcare among many others, which significantly facilitate our daily lives. Considering the unprecedented volumes of data generated by IoT, it will not be possible to satisfy the throughput, delay and security requirements of these use cases with legacy network infrastructures. 5G networks that rely on software-defined networking (SDN) and network function virtualization (NFV) for resource management will be a key enabler for the future's ubiquitous IoT.

Recent years have witnessed an increasing use of machine learning (ML) techniques in network intrusion detection systems (IDS) to detect various types of attacks. Despite the importance of realistic network traffic data for effective model building, most of the existing research in IoT intrusion detection has used datasets that were generated for legacy networks without IoT traffic. This is mostly due to the lack of publicly available datasets that include IoT traffic, except the recently released Bot-IoT [1]. Also, to the best of our knowledge, there is no publicy available dataset specifically for SDN-based IoT environments. The network traffic characteristics of IoT and SDN are quite different from those of legacy networks, therefore, using models trained with legacy network data might lead to inaccurate classification results. Furthermore, it is crucial for an IDS to retrieve features in real time for effective attack detection and mitigation. Existing public datasets have been created by processing pcap files and there is no guarantee that all features they include can be retrieved in real time.

In order to overcome the shortcomings of existing intrusion detection datasets, in this paper we introduce a novel dataset for IoT environments managed by SDN, which includes both normal traffic and different types of attack traffic.

This study is partially supported by Turk Telekom within the framework of 5G and Beyond Joint Graduate Support Programme coordinated by Information and Communication Technologies Authority.

978-3-903176-31-7 © 2020 IFIP

II. Preliminaries

A. Software-defined networking (SDN)

SDN is a networking paradigm that separates the control plane from the data plane unlike traditional networks, where switches perform both forwarding and routing. The controller, which has the global view of the system, manages the network and makes routing decisions. The forwarding application runs on the controller and creates forwarding rules using the routing information. The controller installs these forwarding rules called "flow entries" into the switches. Switches in SDN are simple devices that are only responsible for forwarding the packets. They maintain one or more flow tables that contain flow entries.

When a switch receives a packet, it checks if the headers of the packet match any flow entry in its flow tables. In order to match a flow entry, all of the match fields must be the same as the headers of the packet. If a packet does not match any flow entry, it results in a table miss event. The switch buffers the packet and sends the header of the packet to the controller in a packet called *packet_in* message. The controller processes the packet and sends a *packet_out*

message containing the determined action upon the packet that triggered the table miss event. The packet is forwarded or dropped according to the *packet_out* message. The controller might also decide to install flow rules into the switches along the way by sending *flow_mod* messages [2]. When the switch receives the flow_mod message, the flow entry remains in pending_add state until a packet matches it.

The flexible nature of SDN and its ability to quickly reconfigure forwarding rules offers great security advantages for IoT over legacy network architectures by enabling the detection and isolation of malicious traffic in real time.

III. Related Work

With the increasing usage and future use cases of SDN and IoT in recent years, serious research efforts have been dedicated to the security of SDN-based IoT networks.

The most commonly used datasets for SDN-based intrusion detection research and their explanations are given below.

1) **NSL-KDD** dataset [3] is the enhanced version of the famous KDD CUP'99 dataset [4] used for intrusion detection research for the past two decades. The purpose of this dataset was to address some limitations of KDD CUP'99. Some of the duplicated records in the training and test sets were removed. Records in the training and test datasets were reduced and balanced. However, this dataset does not reflect the behavior of IoT networks.

2) **CAIDA** dataset [5] records are anonymized network traces. Payload of the packets were removed and headers of the packets were anonymized. The first drawback of the CAIDA dataset is that it only contains DoS attacks. The second drawback is that records only contain header information and additional features using these fields were not generated.

3) **UNSW-NB15** dataset [6] was created using IXIA traffic generation tools. The dataset contains 49 features and 9 different attack types: fuzzers, anaylsis, backdoors, DoS, exploits, generic, reconnnaissance, shellcode and worms. The main disadvantage of the dataset is that some attack categories have limited number of samples. For example, worms have 174 records, which is not sufficient for most machine learning algorithms to learn accurate models.

4) **CICIDS2017** dataset [7] was created by the Canadian Institute of Cybersecurity. Authors used their B-Profile system to create realistic normal traffic. The dataset contains 6 types of attacks: DoS, port scanning, botnet, brute force, web attacks and infiltration.

5) **Bot-IoT** dataset [1] is a new dataset created in 2018. The dataset contains both IoT and normal network user traffic. The dataset contains 46 features and 6 types of attacks: service scanning, OS fingerprinting, DoS, DDoS, keylogging and data theft. However, this dataset is not balanced and some attack types contain many less records compared to others. There are 118 records for data theft and 1469 for keylogging, which is not sufficient for most machine learning algorithms.

The abovementioned datasets have been used for most of the research in intrusion detection systems for SDN. However, none of these datasets have been created in an SDN environment, therefore they do not reflect SDN behavior. These datasets have been created processing pcap files with various tools. Therefore, there is no guarantee that the SDN controller can retrieve all of the included features in real time. Also none of the datasets, except Bot-IoT, contains IoT traffic. Although some researchers [8] [9] created their own SDN datasets, their data only contained DDoS traffic and was used for testing their ML models. To the best of our knowledge, there is no publicly available SDN dataset containing different types of attacks and normal traffic for IoT so far.

IV. Proposed SDN Datasets

Our SDN datasets were created using a similar topology, features and packet sending rates for benign traffic to the Bot-IoT dataset [1]. Benign traffic was generated and recorded both with and without presence of malicious traffic. 5 different attack types were implemented with various targets and configurations: DoS, DDoS, port scanning, OS fingerprinting and fuzzing. We have created two SDN datasets. Their only difference is the number of simulated IoT devices. IoT networks are dynamic networks and even though the general topology stays the same most of the time, the number of connected IoT devices change from time to time. The purpose of our second SDN dataset is to test the performance of attack detection models trained using the first dataset in a dynamic IoT environment. That way we can see how our detection algorithm will be affected from changing the number of IoT devices and how often our model should be updated.

Our SDN application collected flow entries from the switches at regular intervals and crafted features during traffic generation. All of the features in our datasets can be obtained using an SDN application and detection models can be used without affecting the network performance in real time.

A. Testbed overview

We have simulated our IoT network using the Mininet tool [10]. Our testbed setup is similar to that of Bot-IoT and the setup of the first dataset is shown in Figure 1. In addition, our network is managed by an ONOS controller [11]. During data generation, the server exchanged huge amounts of data with the other two hosts in the system continously. In the first dataset, five hosts in the network simulated IoT services and sent small amounts of data to the server periodically. In the second dataset, initially ten IoT devices were simulated. Two of them were disconnected after a predetermined time. Four attacker hosts targeted the server and IoT devices. All hosts and ONOS controller were connected to an Open vSwitch.

B. Benign traffic

We have examined the normal records in the Bot-IoT dataset and created our normal traffic using similar packet sending rates, packet sizes and transport layer protocols. Bot-IoT dataset has only 9543 records for normal traffic, therefore it was not hard to analyze and replicate the normal traffic. We have generated benign traffic both with and without presence

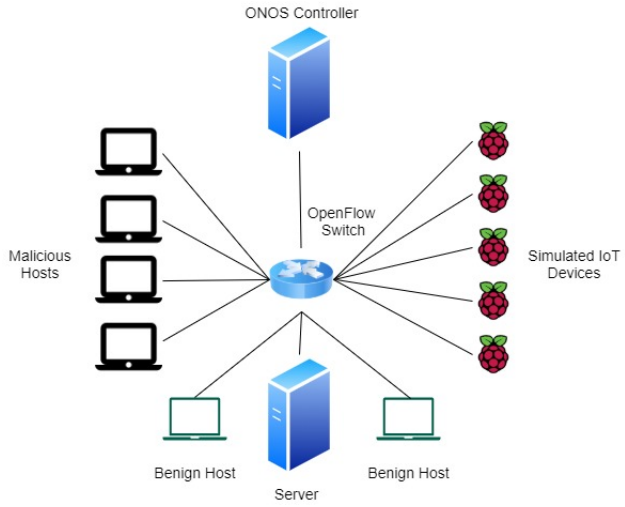

Fig. 1: Testbed Setup.

of attack traffic. Without attack scenarios, we have recorded normal traffic 20 times and saved these records into different csv files. Each recording period lasted 30 to 35 minutes. In the first 15 recordings, we generated both background traffic and IoT traffic. In the last 5 recordings, only IoT traffic was generated. During attack scenarios, both background traffic and IoT traffic were generated and recorded. Recording count was 10 to 12 and duration of the recording was 30 to 45 minutes depending on the attack scenario. Each recording was saved into a different csv file.

1) Background traffic: For the background traffic, two hosts in the network sent packets to the server constantly. One of the hosts used TCP and the other one used UDP. Packet sending rates and sizes were similar to the Bot-IoT dataset's normal traffic. For TCP traffic, we have used 3 different packet sending rates and 4 different packet sizes. For UDP traffic, we have used 2 different packet sending rates and 2 different packet sizes. Packet sending rates and packet sizes were selected by our packet generation script at the start with the probability proportional to the observation count in the Bot-IoT dataset. For example, the probability of selecting a sending rate of 80 packets per second and a packet size of 1000 bytes for TCP packets was 15.8%.

2) IoT traffic: In the Bot-IoT dataset, almost 6000 of the 9543 normal records used UDP and sent only 1 or 2 packets with less than 100 bytes. IoT devices were simulated using Node-red tool and simulated services were weather station, smart fridge, motion activated lights, remotely activated garage door and smart thermostat. All of these services send small amounts of data periodically. We also used these byte counts as packet sizes when we simulated IoT devices in our network. We have used 6 different packet sizes and sent one or two packets each time. In the case of sending two packets, packets were sent back to back. Simulated IoT devices sent packets to the server periodically. ONOS's default flow rule timeout is 10 seconds. If a flow entry does not match any incoming packet for the 10 seconds, the flow entry is deleted. Packet sending periods were also selected randomly from 12 seconds to 15 seconds by our script, which caused every incoming IoT packet to trigger new flow rule installation. Due to the random selection of waiting intervals, the number of flow entries that belong to the IoT devices in the switch was changing dynamically. We used TCP instead of UDP for simulated IoT hosts. The reason is that flow rules are installed after the packet that triggers flow rule installation was forwarded by a packet_out message as explained in the preliminaries section. If we used UDP, we would not be able to collect the statistics of the first packet (duration, byte and packet count). We have used TCP and the first packet sent was the SYN message, which is the same for all TCP connections. Therefore, not having SYN packets' statistics was not important.

C. Malicious traffic

Four attacker hosts in our network launched the cyber-attacks targeting the server or simulated IoT devices. We performed 5 different attack types: DoS, DDoS, port scanning, OS fingerprinting and fuzzing. Each attack type was performed with different configurations 10 to 12 times depending on the attack type and each configuration was saved into different csv files. The attack types and details regarding their implementation are explained below.

1) **DoS:** One attacker host was used to perform DoS attacks targeting the server or an IoT device using the hping3 tool [12]. We recorded DoS traffic 12 times. Each recording lasted 30 to 45 minutes. In two recordings, the server was targeted with a TCP SYN flooding attack using spoofed IP addresses and source port numbers. In the other two recordings, the server was targeted with a UDP flooding attack using spoofed IP addresses and source port numbers. In one of the recordings, the server was targeted with a TCP SYN attack without spoofing. In one of the recordings, the server was targeted with a UDP flooding attack without spoofing. During the attacks without spoofing, all of the packets coming from the attacker passed over the same flow entry and new flow rule creation was not triggered. In the remaining 6 recordings, the same attack scenarios were performed targeting one of the IoT devices.

Four different packet sending rates were used: 4000, 6000, 8000 and 10000 packets/second. Four different packet payload sizes were used: 0, 100, 500 and 1000. All of the combinations of these packet sending rates and payload sizes were used.

2) **DDoS:** Four attacker hosts were used to perform DDoS attacks targeting the server or an IoT device using the hping3 tool [12]. We recorded DDoS traffic 12 times with the same scenarios as the DoS attacks. Each recording lasted 30 to 45 minutes. We also used the same packet sending rates and packet sizes as the DoS attacks.

3) **Port scanning:** One attacker host was used to perform a port scanning attack targeting the server or an IoT device using nmap [13]. Nmap has two modes for port scanning. If you do not specify the port numbers which should be scanned, nmap scans the first 1024 ports. These port numbers are called well-known port numbers and reserved for the most commonly used services. We have recorded port scanning traffic 10 times. Each

recording lasted 30 to 45 minutes. In two recordings, nmap was used to scan all of the ports of the server. In the other two recordings, all of the ports of an IoT device were scanned. In the other three recordings, the first 1024 port numbers of the server were scanned. In the last three recordings, the first 1024 port numbers of an IoT device were scanned.

4) **OS fingerprinting:** One attacker host was used to perform OS fingerprinting targeting only the server using the nmap tool [13]. We recorded OS fingerprinting traffic 10 times. Each recording lasted 30 to 45 minutes.

5) **Fuzzing:** We used boofuzz [14] for fuzzing attacks. Boofuzz is installed as a Python library, which is used to build fuzzing scripts. We implemented HTTP and FTP fuzzer scripts. HTTP and FTP server code were running on our server. One of the attacker hosts were used to perform HTTP and FTP fuzzing attacks against the server. We have recorded fuzzing attacks 10 times. In the first five recordings, FTP fuzzing was used. In the last five recordings, HTTP fuzzing was used.

D. Flow collection and feature generation

Most of the existing intrusion detection datasets in the literature have been created by processing pcap files after traffic generation. Our purpose was to create a dataset specific to SDN and use this dataset to prevent attacks using the SDN controller in real time. Therefore, we developed an SDN application for retrieving flow entries from the network switches every 1 second and creating our features for each flow. Our application labeled each flow using the ports on which hosts are connected to the switch and wrote the features into csv files. Every recording session of every attack type was saved into different csv files.

We tried to create all of the features in the Bot-IoT dataset. Our SDN application only used the flow entry fields pulled from the switches, hence all of the features in our dataset are SDN-specific and can be calculated using an SDN application. We were not able to create features in the Bot-IoT that are not specific to SDN environments (e.g. argus sequence number). The datasets contain 33 features. Feature names and descriptions are given in Table I.[1]

Normal traffic was recorded during all attack scenarios and without attack scenarios. DoS and DDoS attacks with spoofed IP addresses created huge numbers of duplicate flow entries. Therefore, we limited saved record count to 100 every time flow entries were pulled from the switch for DoS and DDoS attacks. Port scanning and OS fingerprinting attacks also created huge numbers of flow entries, but we did not limit them because created packets were not duplicates. Our first SDN dataset contains 27.9 million entries. Our second dataset was created using the same scenarios. The only difference was the higher number of IoT devices. The second dataset contains 30.2 million entries. Tables II and III show the entry counts and percentages for every traffic category in the datasets.

[1]The dataset is available at https://github.com/AlperKaan35/SDN-Dataset.

TABLE I: Features of the created SDN datasets

Feature	Description
srcMac	Source MAC address
dstMac	Destination MAC address
srcIP	Source IP address
dstIP	Destination IP address
srcPort	Source port number
dstPort	Destination port number
last_seen	Record last time
Protocol	Textual representation of network protocol
proto_number	Numerical representation of network protocol
Dur	Record total duration
Mean	Average duration of aggregated records
Stddev	Standard deviation of aggregated records
Min	Minimum duration of aggregated records
Max	Maximum duration of aggregated records
Pkts	Total count of packets in transaction
Bytes	Total number of bytes in transaction
Spkts	Source-to-destination packet count
Dpkts	Destination-to-source packet count
Sbytes	Source-to-destination byte count
Dbytes	Destination-to-source byte count
Srate	Source-to-destination packets per second
Drate	Destination-to-source packets per second
Sum	Total duration of aggregated records
TnBPSrcIP	Total number of bytes per source IP
TnBPDstIP	Total number of bytes per destination IP
TnP_PSrcIP	Total number of packets per source IP
TnP_PDstIP	Total number of packets per destination IP
TnP_PerProto	Total number of packets per protocol
TnP_Per_Dport	Total number of packets per destination port
N_IN_Conn_P_SrcIP	Number of inbound connections per source IP
N_IN_Conn_P_DstIP	Number of inbound connections per destination IP
Attack	Attack or not
Category	Traffic category

TABLE II: Distribution of records in the first SDN dataset

Category	Size	%
Normal	1.67M	5.99
DoS	793K	2.84
DDoS	192K	0.67
Port Scanning	20.68M	74.08
OS and Service Detection	3.39 M	12.15
Fuzzing	1.18M	4.24

V. Conclusion

In this paper we introduced a novel dataset for SDN-assisted intrusion detection in IoT environments. The created dataset consists of a high number of data records consisting of traffic modeling various IoT device types and different attacks. To the best of our knowledge, this is the first publicly available SDN-specific intrusion detection dataset for IoT, hence will serve as an important resource for IoT security research.

TABLE III: Distribution of records in the second SDN dataset

Category	Size	%
Normal	2.67M	8.84
DoS	495K	1.67
DDoS	182K	0.60
Port Scanning	22.44M	74.23
OS and Service Detection	3.39 M	11.20
Fuzzing	1.05M	3.48

REFERENCES

[1] 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," *CoRR*, vol. abs/1811.00701, 2018.

[2] T. Alharbi, S. Layeghy, and M. Portmann, "Experimental evaluation of the impact of dos attacks in SDN," Proceedings of the 27th International Telecommunication Networks and Applications Conference, ITNAC 2017, Melbourne, Australia, November 22-24, pp. 1–6., 2017.

[3] 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*, pp. 1–6, 2009.

[4] S. Hettich and S. Bay, "The uci kdd archive [http://kdd.ics.uci.edu]. irvine, ca: University of california," *Department of Information and Computer Science*, vol. 152, 1999.

[5] Center for Applied Internet Data Analysis. https://www.caida.org/data/.

[6] N. Moustafa and J. Slay, "Unsw-nb15: A comprehensive data set for network i intrusion detection systems (unsw-nb15 network data set)," Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS), pp. 1–6., IEEE, 2015.

[7] A. H. L. Iman Sharafaldin and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," Proceedings of 4th International Conference on Information Systems Security and Privacy (ICISSP), Portugal, pp. 108–116, January 2018.

[8] J. Galeano-Brajones, J. Carmona-Murillo, J. F. Valenzuela-Valdés, and F. Luna-Valero, "Detection and mitigation of dos and ddos attacks in iot-based stateful sdn: An experimental approach," *Sensors*, vol. 20, p. 816, Feb 2020.

[9] N. Ravi and S. M. Shalinie, "Learning-driven detection and mitigation of ddos attack in iot via sdn-cloud architecture," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3559–3570, 2020.

[10] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, Hotnets-IX, (New York, NY, USA), pp. 19:1–19:6, ACM, 2010.

[11] ONOS. https://www.opennetworking.org/onos/.

[12] Hping3. http://www.hping.org/hping3.html.

[13] Nmap. https://nmap.org/.

[14] Boofuzz. https://boofuzz.readthedocs.io/en/stable/.