GRADIENT BOOSTING ALGORITHM FOR EARLY DETECTION OF UNKNOWN INTERNET OF THINGS DEVICES

*Vian A. Ferman¹

Mohammed A. Tawfeeq¹

1) Computer Engineering Department, College of Engineering, Mustansiriyah University, Baghdad, Iraq

Abstract: The pervasive availability of the Internet of Things (IoT) markets lures targets for cyber-attacks since most manufactured IoT devices are usually resource-constrained devices. The first powerful line of IoT network protection from these vulnerabilities is detecting IoT devices especially the unauthorized ones by utilizing machine learning (ML) algorithms. Actually, it is so difficult or even impossible to find individual unknown IoT devices during the setup phase but, knowing their manufacturers is a matter to be deliberate. In this paper, a new method based fingerprints generation is introduced to detect the connected devices in the setup phase. Fingerprints for 21 different IoT devices are generated using devices' network traffic. The whole produced fingerprints of devices are divided into four groups according to their manufacturers or fingerprints similarity proportion. Gradient Boosting Algorithm is applied to achieve the identified purposes. The proposed method is considered as a preparatory study for early detection of unauthorized. The performance evaluation for the proposed method was calculated based on two metrics: Identification accuracy and F1-score. The average identification accuracy rate was around 98.65%, while the average F1-score was about 99%.

Keywords: *Device Fingerprint, Gradient Boosting Algorithm, Internet of Things (IoT), Machine Learning, Network Traffic.*

1. Introduction

Internet of Things (IoT) is defined as a distributed and interconnected network of embedded systems which are communicated through either wired or wireless communication network technologies. It is also regarded as the network of things or physical objects empowered with limited communication capabilities,

computation, and storage as well as it is embedded with electronics (e.g. sensors and actuators), application, and network connectivity that giving these objects the ability to collect, exchange, and sometimes process data [1]. IoT is globally expanding, offering diverse benefits in almost every aspect of human life [2]. It is expected that the campuses' and future enterprises' networks will be instrumented with a massive number of smart devices to provide remote control, surveillance, security, entertainment, and powerful management for smart cities and industries [3].

Along with the benefits, this rapid development and integration of IoT in addition to the heterogeneity of the connected devices cause the security problem of IoT to be an urgent threat for this age. Once the IoT vulnerabilities are exploited by attackers, it will give them the ability to control the device, privacy leakage of users, and posing other security concerns like IoT Mirai botnet and launch some types of attacks on IoT network infrastructure which lead to network congestion [4], [5]. Although the embedded security modules are produced by many vendors on the markets, many attackers continue holding a dominant position due to the unprecedented amount of daily production and malware types [6]. Therefore, knowing which IoT devices are

*Corresponding Author: egma018@uomustansiriyah.edu.iq

associated with the network became an important topic to think about, especially the unknown devices.

The classification of IoT network traffic is a significant aspect of administration systems and current network management since it can be used to retrofit the IoT network with devices that can offer substantial and smart functionalities. There are distinct approaches of network traffic classification such as payload-based, flow statistics-based, and port-based. The method of payload-based utilizes deep packet inspection (DPI) of the raw payload and looks for the well-known patterns inside the network packets. DPI produces a high overhead and it probably fails when the network traffic is encrypted [7], [8]. Moreover, the flow statistics-based approach relies on packet header information (e.g. the transmitted bytes, TCP window size, interarrival times, etc.) [8], [9]. Whereas, the port-based approach base on extracting port information but is an unreliable technique because many services do not use well-known ports or may use ports that are utilized by other applications [8].

Many researchers focused on identifying IoT devices during normal operation without making attention to isolate or predict unauthorized ones. In [10], a two stages classifier is trained. Firstly, network traffic is classified to distinguish between IoT and non-IoT smart devices. Then, IoT devices are identified according to their associated class. Also, a two stages classifier is used [11] to classify 21 types of IoT devices by utilizing a network and statistical features. While in [12], ten machine learning (ML) algorithms are applied to classify 128 events of 39 distinct IoT devices.

Furthermore, few researchers took unknown devices' isolation or detection into consideration. In [2], a Random Forest (RF) algorithm was applied to detect the type of 17 unauthorized IoT

devices. The devices were grouped into nine types and each time eight of them were regarded as devices' types within the white list and the remaining as unknown. Based on the identification of 20 successive sessions, the average accuracy of identification of the unknown devices' type was 96%. Meanwhile, deep learning was applied using traffic payload data to identify nine IoT devices in [13]. Each time only one device considered unauthorized and the rest as known devices. Moreover, a method presented [14] to automatically identify unseen and new devices using the rich information of traffic flows. Deep learning was applied to classify fifteen devices into four types and achieves 74.8% as an average accuracy.

Besides, some researchers looked forward to identifying devices during the setup phase and also making attention to predict unknown ones. In [15], RF was applied to detect the type of 27 devices that achieved an accuracy of about 81.5%. In [16], three ML models are applied to also identify the type of ten devices and achieved an accuracy of about 93.75%. In [17], a locality-sensitive for IoT fingerprints was presented to detect devices when they joined to IoT network. It achieved 90% as an average recall and 93% as an average precision.

According to the former studies, a few researchers are concentrated on detect unknown devices especially in the startup phase which regards the best time to detect such devices to make an early decision about their traffic. Knowing unknown traffic for each individual device and even for each device type is regarded as unpractical with the diversity of IoT devices because it is impossible to cover all devices within a model especially since new devices are introduced every day. So, knowing unknown devices according to their manufactures is a matter of this paper. The contributions of this work are:

- 1- The traffic of seven different testbed IoT devices (related to four different manufacturers) is collected during the startup phase.
- 2- The dataset is enriched with fourteen distinct devices taken from public traffic: captures_IoT-Sentinel [15], [18].
- 3- Fingerprints are created for all devices (each with 13 in length) utilizing TCP and UDP data payload protocols packets.
- 4- Group all devices into four groups (each with 5-6 devices) according to their manufactures or similarity proportion.
- 5- To equalize all groups' fingerprints number, Gaussian noise is utilized.
- 6- Gradient boosting classifier (GBC) is applied to detect devices' fingerprints according to the created groups.

The remainder of the paper is structured as follows: traffic collection and analysis are introduced in section 2. The proposed method of creating and identifying fingerprints of devices are presented in section 3. The identification results and discussion are clarified in section 4. Finally, the conclusion of this work is presented in section 5.

2. IoT Network Traffic Collection and Analysis

To connect the testbed IoT devices to the network, an access point is required. Furthermore, to monitor, analyze, and collect devices' network traffic, the access point should have all these capabilities. Therefore, Raspberry Pi 3 Model B+ is a good choice to be configured to work as an access point. After configuration, Wireshark Network Protocol Analyzer is installed. To check other devices' traffic rather than the used testbed, enrich the proposed system with the public traffic (captures_IoT-Sentinel) of only devices with wireless technology. Table 1 shows all tested IoT devices.

Table 1. Tested IoT devices.

Manufacture		Device name	
1	SonoFF (4)	Bulb, Power Strip, Power Plug, Smart Switch	Testbed devices
2	TEKIN (1)	Plug	
3	Aswar (1)	Camera	
4	Google (1)	Chromecast	
5	D-Link (5)	D-LinkCam, D-LinkSensor, D-LinkSwitch, D-LinkSiren, and D-LinkWaterSensor.	Online traffic
6	WeMo (3)	WeMoSwitch, WeMoInsightSwitch and WeMoLink	
7	TP-Link (2)	TP-LinkPlugHS100 and TP-LinkPlugHS101	
9	Ednet (1)	EdnetGateway	
10	Withings (1)	Withings	
11	Fitbit Aria (1)	Aria	
12	Osram (1)	Lightify	

When any device connects to the access point through wireless technology, a set of protocols are generated began with Extensible Authentication Protocol over LAN (EAPOL) packets then DHCP as well as other protocols packets (e.g. DNS, ICMP, IGMP, etc.). Actually, depending on these protocols aren't enough to create strong fingerprints since most devices share some of these protocols.

After these protocol packets, the connected devices are started to exchange data predominantly using protocols like TCP, UDP, or both. While analyzing the collected traffic, it is found very small differences in the characteristics of the first generated TCP session and sometimes the first UDP packets. These few differences not only within the same device but sometimes within devices of the same manufacture. That's mean the convergence is available on some features of devices within the same manufacturer. Fig. 1 shows TCP and UDP DATA packet counts of IoT device traffic while exchanging data for the first time. The packet

number is computed until the first TCP session is closed (TCP with FIN flag). As shown, four D-Link devices exchange packets with a remote device using TCP with HTTP protocols (port number = 80) and just D-LinkCam produce TCP with HTTPS protocols (port number = 443). Furthermore, all SonoFF devices TCP are with HTTP protocol and all with eleven packets. Besides, there are some differences within WeMo packets features. TEKIN- Plug as well as

EdnetGateway and TP-Link devices all produce TCP packets but with almost dynamic ports. Withings and Aria produce HTTP packets while Lightify generates only UDP-DATA packets. In general, these packets may be changed in number each time the traffic is collected, but here take the one which more appeared. Chromecast and Aswar_Camera are both unstable not only in packets number but sometimes in the used protocol type.

Figure 1. TCP and UDP DATA packets counts of IoT devices traffic while exchanging data for the first time

3. Proposed Method

This section introduces the fingerprint generation method and how devices' fingerprints are grouped. Next, a brief overview of the GBC algorithm used to classify these fingerprints according to the proposed groups.

3.1. Fingerprint Generation Method

Since EAPOL packets are the first packet generated when any wireless device initializes to connect to the access point, it is a suitable option if taken as fingerprint generation control. So, the first TCP or UDP-DATA packet is taken and the related ports are saved to gather all TCP and UDP having these ports.

Hereby, other packets are neglected. A 2D matrix with an $n \times 8$ dimension is created, where n refers

to the variable length of the matrix. The first three columns for TCP protocol types (HTTPS, HTTP, TCP with dynamic ports). The next three columns are for TCP packet length, TCP segment length, and TCP window size. The last two columns are for UDP-DATA packets and their length respectively. The matrix-fill method sets 1 for the three types of TCP plus UDP-DATA if any of them exist and otherwise sets 0. While the values of other features are put as they come. When TCP with FIN flag is found the process of gathering those features is stopped. Sometimes, TCP packets with dynamic ports don't introduce the FIN flag, so only 20 packets are considered. Moreover, only a few devices generate UDP-DATA at the setup time, therefore only five

packets are taken because it is enough to distinguish them.

After the process of gathering features is finished, the generated matrix is converted to a vector with 13 in length (Fingerprint_vector = $[f_0, f_1, f_2, \dots, f_{12}]$) as follows:

- 1- Count number of 1 for the TCP protocol type's columns as well as UDP-DATA column.
- 2- Compute MIN, MAX, and AVERAGE for both TCP packet length and TCP window size columns.
- 3- Compute only average for the TCP segment length column.
- 4- Compute MIN and MAX for the UDP-DATA length column.

Table 2 shows a sample of Fingerprint_vector related to SonoFF-Plug. Fig. 2 shows a sample of five statistical operations related to TCP features of each device fingerprint (4-8 columns in Fingerprint_vector). The figure explains how

could create groups so, four groups are created namely: D-Link is for all D-Link devices, SONOFFTE for all SONOFF and TEKIN-Plug, WETPCAST for (WeMo devices, TP-Link devices, and Chromecast), and OTHERS group is for all other devices.

Table 2. Sample of the generated Fingerprint_vector of SonoFF-Plug

Index	Feature name	value
0	HTTPS count	11
1	HTTP count	0
2	TCP with dynamic ports count	0
3	TCP Min packet length	54
4	TCP Max packet length	782
5	TCP average packet length	212.4545
6	TCP average segment size	289.1667
7	TCP Min window size	4744
8	TCP Max window size	28944
9	TCP Average window size	15367.55
10	UDP-DATA count	0
11	UDP-DATA Min length	0
12	UDP-DATA Max length	0

Figure 2. Sample of five statistical operations related to TCP features of each device fingerprint

The small differences between those five features as well as the remaining features are qualified to be distinguished by ML. The idea of creating OTHERS group with different features is to take

more probabilities of unknown devices' fingerprints.

Since the public traffic is collected as 20 files for each device and the inequality of the created groups, Gaussian noise (GN) is applied with

mean = 0 and standard deviation = 0.1. The probability density function (PDF) of GN is shown in Equation (1), where μ refers to the mean, while σ refers to the standard deviation.

$$PDF_{GN}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (1)$$

Noise is added after the dataset has been standardized to make a strong impact. The

uniformity equation for each sample (x) is shown in Equation 2.

$$standardize(x) = \frac{x - \mu}{\sigma} \quad (2)$$

After noise addition, 125 fingerprints are within each group. Fig. 3 depicts the schematic diagram of the fingerprints generation, while Fig. 4 shows the histogram of fingerprints' features for the whole system.

Figure 3. Schematic diagram of the fingerprints generation.

Figure 4. Histogram of fingerprints' features (each feature is named by its index as listed in Table 2)

3.2. Gradient Boosting Classifier (GBC)

GBC is one of the flexible ML algorithms which don't need data normalization. It can be defined as a group of ML algorithms that combine several weak learning models to create a strong and effective predictive model [19]. Decision trees are regarded as a weak learner so it is usually used with GBC. In addition to the weak learning, the other two main components of this model are: Loss function and additive model. The loss function's purpose is to determine how good the model is at future observations based on the available dataset. The additive model is the iterative and sequential method of adding weak learners. After each iteration, it should get closer to the final model which means each iteration should decrease the value of the loss function. At first, GBC finds the optimal initial values and then the pseudo-residuals. Equation 3 shows the prediction function of x , where r refers to the learning rate, γ is a multiplier, and $h_m(x)$ refers to the regression tree which represents the additional model [11].

$$f_m(x) = f_{m-1}(x) + r \times \gamma \times h_m(x) \quad (3)$$

4. Result and Discussion

The performance evolution of the GBC model is based on computed F1-score and fingerprints identification accuracy as shown in equations below, where TP represents true positive, TN represents true negative, FP represents false positive, FN represents true negative [20].

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

$$Precision = \frac{TP}{TP + FP} \quad (5)$$

$$F1 - SCORE = \frac{2 \times Recall \times Precision}{Recall + Precision} \quad (6)$$

$$Identification Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (7)$$

The fingerprints dataset was splitted into training data (60%) and testing (40%) and the chosen learning rate was 0.2. The high ratio of the testing dataset is taken to show the system identification ability and knowing how the correctness of dividing the fingerprints into groups is. It shows

that the average identification accuracy about 98.65% for ten consecutive executions as well as it achieved an F1-score of around 99%. Fig. 5 shows the performance evaluation of all groups. As shown in Fig. (5.a), the precisions of the D-link are stable (100%). When precision is equal to 100% that means TP is equal to 100% which means all observations actually belong to that group. The minimum ratio of recalls of the D-link group is 98% that means a few samples of the D-link group regarded as another group. Hereby, the average F1-score and accuracy of the D-link group are 99.4% and 99.7% respectively.

While Fig. (5.b) shows the performance evaluation of OTHERS group. Although, the devices are different in this group and noise is added, the recall ratio was 100% which means all devices within this group identified well. The verity of devices in this group leads to misclassifying some samples of the other groups to be within OTHERS group. However, the minimum precision ratio was 90%. The average F1-score and accuracy of the OTHERS group are 98.3% and 99.1% respectively.

Moreover, Fig. (5.c) shows the performance evaluation of the SONOFFTE group. All devices within the group identified well with precision equal to 100% except one time which achieved 95%. Recall ratios were between 88% and 100% which means some samples are identified incorrectly. The average F1-score and accuracy of the SONOFFTE group are 98.8% and 99.4% respectively.

Finally, Fig. (5.d) shows the performance evaluation of the WETPCAST group. Although instability of both precision and recall ratios, it still achieves good ratios which about 98.4% and 98% as an average precision and recall respectively. Table 3 presents an average performance evaluation of all groups. Fig. 6 shows the Confusion matrix of the generated groups. As shown, there is only one mistake in classification within all groups except the OTHERS group.

Fig. 7 shows Fingerprints' Features importance which is computed by the GBC method. As shown, TCP Min window size is regarded as the most important feature, while UDP-DATA features tend to be zero important features.

Figure 5. Performance evaluation of all groups within ten consecutive times

Table 3. Average performance evaluation

Groups	precisio n	Recal l	F1- scor e	Accurac y
D-LINK	0.998	0.99	0.994	0.997
OTHERS	0.971	0.996	0.983	0.991
SONOFFT E	0.995	0.98	0.988	0.994
WETPCAS T	0.984	0.98	0.982	0.991

Figure 6. Confusion matrix of the generated groups **Figure 7.** Fingerprints' Features importance

5. Conclusion

This work regards as a qualification study for early identifying unknown devices. Usually, the known devices are identified with only one machine learning phase and all unknown devices

are isolated under an unknown label without giving any details about them. So, the proposed method is for using as a second phase to identify only unknown devices according to their manufactures. Utilizing the first TCP session properties and UDP-DATA for creating fingerprints lead to achieving about 98.65% overall accuracy. Although UDP-DATA features are the less important features in the case of this work as well as most tested devices that produce UDP-DATA are grouped in OTHERS group, it may be more important to detect unknown devices which deal with UDP-DATA packets as OTHERS group. However, the achieved accuracy, it isn't mean the different devices within same group regard as from one manufacture since with supplying more device the small differences qualified to identify each with its manufacture. Classification errors are expected due to grouping of different devices, adding Gaussian noise, or selecting a large part of the data for testing (40%). In the future, more devices are being considered and attempt to analyze device data with similar manufacturers as well as expand the system with data of IoT devices infected by malware and cyber-attacks.

Acknowledgements

We would like to express our special gratitude to Mustansiriyah University (www.uomustansiriyah.edu.iq) for supporting and giving us a special opportunity to complete this work.

Conflict of interest

The authors confirm there is no conflict of interest in publishing this article.

6. References

1. F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain. (2020). "Machine learning in IoT security: Current solutions and future challenges". IEEE Communications

- Surveys & Tutorials, Vol. 22. No. 3, pp. 1686-1721.
2. Y. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N. O. Tippenhauer, J.D. Guarnizo, and Y. Elovici. (2017). *"Detection of unauthorized IoT devices using machine learning techniques"*. arXiv preprint arXiv:1709.04647.
3. A. Sivanathan, H. H. Gharakheili, and V. Sivaraman. (2018). *"Can we classify an iot device using tcp port scan?"*, in 2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS). IEEE, pp. 1-4.
4. G. Hu and K. Fukuda. (2020). *"Toward Detecting IoT Device Traffic in Transit Networks"*, in 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC). IEEE. pp. 525-530.
5. H. Guo and J. Heidemann. (2018). *"IP-based IoT device detection"*, in Proceedings of the 2018 Workshop on IoT Security and Privacy. pp. 36-42.
6. J. Bao, B. Hamdaoui, and W.-K. Wong. (2020). *"Iot device type identification using hybrid deep learning approach for increased iot security"*. 2020 International Wireless Communications and Mobile Computing (IWCMC) IEEE, pp. 565-570.
7. O. Salman, I. H. Elhajj, A. Chehab, and A. Kayssi, A. (2019). *"A machine learning based framework for IoT device identification and abnormal traffic detection"*. Transactions on Emerging Telecommunications Technologies, e3743.
8. M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret. (2017). *"Network traffic classifier with convolutional and recurrent neural networks for Internet of Things"*. IEEE Access, Vol. 5, pp. 18042-18050.
9. G. Cirillo and R. Passerone. (2020). *"Packet Length Spectral Analysis for IoT Flow Classification Using Ensemble Learning"*. IEEE Access, Vol. 8, pp. 138616-138641.
10. Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici. (2017). *"ProfilIoT: a machine learning approach for IoT device identification based on network traffic analysis"*, in Proceedings of the symposium on applied computing, pp. 506-509.
11. A. Hameed and A. Leivadeas. (2020). *"IoT traffic multi-classification using network and statistical features in a smart environment"*, in 2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD). IEEE. pp. 1-7.
12. B. Charyyev and M. H. Gunes. (2020). *"Iot event classification based on network traffic"*, in IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE. pp. 854-859.
13. J. Kotak and Y. Elovici. (2020). *"IoT device identification using deep learning"*, in Conference on Complex, Intelligent, and Software Intensive Systems. Springer, Cham. pp. 76-86.
14. L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Z. Yang. (2018). *"Automatic device classification from network traffic streams of internet of things"*, in 2018 IEEE 43rd conference on local computer networks (LCN). IEEE. pp. 1-9.
15. M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. R. Sadeghi, and Tarkoma, S. (2017). *"Iot sentinel: Automated device-type identification for security enforcement in iot"*, in 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS). IEEE. pp. 2177-2184.

16. W. Cheng, Z. Ding, C. Xu, X. Wu, Y. Xia, and J. Mao. (2020). "*RAFM: A Real-time Auto Detecting and Fingerprinting Method for IoT devices*", in Journal of Physics: Conference Series. IOP Publishing. Vol. 1518, No. 1, p. 012043.
17. B. Charyyev and M. H. Gunes. (2020). "*IoT Traffic Flow Identification using Locality Sensitive Hashes*", in ICC 2020-2020 IEEE International Conference on Communications (ICC). IEEE. pp. 1-6.
18. "Kaggle website". [Online]. Available: <https://www.kaggle.com/drwardog/iot-device-captures>. [Accessed May 28, 2021].
19. N. Ponomareva, T. Colthurst, G. Hendry, S. Haykal, and S. Radpour. (2017). "*Compact multi-class boosted trees*", in 2017 IEEE International Conference on Big Data (Big Data). IEEE. pp. 47-56.
20. A. Subahi and G. Theodorakopoulos. (2019). "*Detecting IoT user behavior and sensitive information in encrypted IoT-app traffic*". *Sensors*, Vol. 19, No. 21, p. 4777.