Detection Cyber Attacks for Smart Environment using Machine Learning

[1] Chaw Su Htwe, [2] Yan Naung Soe*
[1] Faculty of Computer Science, University of Computer Studies, Lashio, Myanmar
[2] Faculty of Computer Systems and Technologies, University of Information Technology, Yangon, Myanmar
{chawsuhtwe, *yannaungsoe}@ucsy.edu.mm

Abstract— **Deployment of smart devices in many situations will make our daily life more comfortable and make it more efficient. These smart devices could also use at smart home, smart city and many other smart systems for the education, organization and so on. However, the growing of malware attacks is targeting smart, IoT devices. After infecting the malware attacks on these devices, they become bots which are controlled by attackers, and these will be targeted to the organizations not only for stealing important information but also for breaking down the network. There are many security mechanisms to protect against cyber attacks based on rule-based detection techniques and cryptographic techniques. The lightweight cryptographic methods can be crushed because of the increment of the hacker's computation capability such as quantum computing. And also, formal rule-based detection could be circumvented by the malware attackers' knowledge. Therefore, the machine learning-based detection scheme is the replacement for the lack of previous detection techniques. The modern cyber attack dataset, N-BaIoT is used for detection cyber attack, especially botnet attack. The experiments are done by CART algorithm and the average accuracy is more than 99%.**

Keywords— Cyber Attacks, Machine Learning, CART, IoT Malware

I. INTRODUCTION

IoT (Internet of Things) has been coming up and our world will become more convenient and more efficient. According to the Cisco Visual Network Index, mobile data traffic will grow at a compound annual growth rate of 47 percent from 2016 to 2021, reaching 49 exabytes per month by 2021 [1]. The higher the growth of mobile and IoT infrastructure, the more challenging of cyber-security occurs. In the attempted attacks against IoT devices over 2016, the average of IoT device was attacked once every two minutes [2]. Cybercriminals' interest in IoT devices continues to grow, and many malware attacks for smart devices picked up to three times in 2017. Kaspersky Lab has collected 121,588 malware samples in 2018 [3].

There are much malware is targeting on IoT device in recent year. Therefore, it is needed to implement an effective detection system. Although there are many previous detection systems, it is not enough to detect all kinds of attacks effectively because of the emergence of the new variant of malware. There are mainly two kinds of detection system, such as misused and anomaly-based detection systems. According to the detection environment, it can be a host-based detection system and network-based detection system.

Misused-based detection is implemented by using the attack signature. Most of the public detection system used that kind of detection system, as in Snort [4], Suricata [5]. Basically, the misused-based detection system can detect the attack based on the attack signatures which is stored in their database. This detection scheme can't detect the unknown attack. Another detection mechanism, anomaly detection systems own unknown attack detection capability. However, this mechanism has a high number of false positive alarm, and it is really difficult for implementing in IoT environment because of the different nature of the IoT devices.

The machine learning-based detection is the possible detection mechanism because it can detect some variant of attacks. Although there are many previous researches were implemented by using machine learning methodologies, most of them are using outdated dataset, mainly KDDCUP 99 [6] and KDD NSL [7]. These datasets have outdated records, and there are no IoT attack records. Therefore, the modern dataset [8], called N-BaIoT is used to build the detection model. This dataset has ten attack classes and one benign class which are captured on the IoT devices by running the malware (botnets) such as Mirai and Gafgyt.

Machine learning is a subset of Artificial Intelligence (AI) in the field of computer science that often uses statistical techniques to give computers the ability to learn with data. It could understand how to program them to learn, to improve automatically with experience and its impact would be dramatic. Decision tree (CART algorithm) is one of the popular machine learning methods to get high detection rates and light processing for cyber attack protection system.

The proposed detection system is intended to develop an effective detection system using machine learning methodology. It is also intended to implement the detection system by using the effective features from public datasets which were captured by running malware samples on real IoT devices. The primary objective of the research is to get an effective detecting system which is the highest detection accuracy for protecting the smart environment.

The paper is organized with five sections. The current challenging of cyber attacks and the detection systems were introduced in this section. The related work of the cyber attack detections researches will be expressed in section 2. Moreover, the background methodology will be presented in section 3. The experiment results will be discussed in section 4. Finally, the discussion will be concluded in section 5.

II. RELATED WORK

Most of the previous IDS researches [9]–[11] used KDDCUP 99 dataset and machine learning algorithms were used for implementing the detection system. Another popular dataset, the variant of KDDCUP 99, named KDD-NSL dataset was also used in IDS researches [10], [13], [14]. However, such datasets are too outdated and not enough the modern attack distribution records. In recent years, as more and more IoT devices are actually deployed, IDS in IoT environments has attracted attention from many researchers and developers. The researches [15], [16] addressed specific types of threats targeting specific devices.

The previous signature-based IDS systems were implemented by many kinds of research [5], [17], [18], it is still needed to get the lightweight proposal for resources constraint IoT devices. The studies [17], [19] showed that formal snort rules are not enough for detection system and their proposals were focused on a traditional network. Although the rule generation proposals [20], [21] were for the IoT environment, their work was based on static analysis.

Some previous researches are based on artificial neural networks (ANN) such as back-propagation algorithm [18][19] and anomaly-based replicator neural network [20]. Another useful algorithm, k-nearest neighbor (KNN) was implemented with anomaly-based [7][12] and hybrid based [11][21]. The random forest (RF) algorithm was also used for anomaly based [12] and hybrid-based [13]. Naïve Bayes (NB) was used in anomaly-based [7] and hybrid-based [11][21]. Most hybrid classifier [7][22] was implemented for the anomaly-based detection system. All of them used the IDS dataset for implementing their detection system. Moreover, this datasets are for traditional network and not for IoT environment.

Although there are many researches for the detection of the cyber attack, it is still needed the lightweight system of IoT environment because of the different nature of IoT devices and resource constraint problem of these devices.

III. METHODOLOGY

The lightweight machine learning algorithm, called CART, was used for the detection model. It is a decision tree algorithm. To evaluate of the proposed detection system, Python libraries, especially scikit learn was used. Firstly, the selected dataset is loaded to the learning program which is implemented by Python language. The 34% of the dataset is used for testing and the remaining part is used for training. And then, the classification results which are based on CART algorithm are extracted by this tool. Finally, the evaluation results are calculated by equation (1).

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Where, TP is true positive which indicates the number of attack class correctly classify, TN is true negative which indicates the number of benign/normal class correctly classify, FP is false positive which indicates the number of attack class wrongly classify and FN is false negative which indicates the number of benign/normal class wrongly classify.

A. CART

The algorithm is based on Classification and Regression Trees by Breiman et al (1984). A CART tree is a binary decision tree that is constructed by splitting a node into two child nodes repeatedly, beginning with the root node that contains the whole learning sample. Classification and regression trees are machine-learning methods for constructing prediction models from data. The models are obtained by recursively partitioning the data space and fitting a simple prediction model within each partition. As a result, the partitioning can be represented graphically as a decision tree. Classification trees are designed for dependent variables that take a finite number of unordered values, with prediction error measured in terms of misclassification cost. Regression trees are for dependent variables that take continuous or ordered discrete values, with prediction error typically measured by the squared difference between the observed and predicted values [22]. The pseudo code for tree construction is shown in Figure 1.

1. Start at the root node.
2. For each X, find the set S that minimizes the sum of the node impurities in the two child nodes and choose the split $\{X^* \in S^*\}$ that gives the minimum overall X and S.
3. If a stopping criterion is reached, exit. Otherwise, apply step 2 to each child node in turn.

Figure 1. The pseudo code for tree construction

CART uses a generalization of the binomial variance called the Gini index, it is shown in equation (2). It stores the sum of squared probabilities of each class, i is from 1 to the number of classes.

$$Gini = 1 - \sum (Pi)^2 \quad (2)$$

This algorithm can handle the missing value. If the dependent variable of a case is missing, this case will be ignored in the analysis. If all predictor variables of a case are missing, this case will also be ignored. If the case weight is missing, zero, or negative, the case is ignored. If the frequency weight is missing, zero, or negative, the case is ignored.

B. Python Libraries

Guido van Rossum created the Python programming language in the late 1980s. In contrast to other popular languages such as C, C++, Java, and C#, Python strives to provide a simple but powerful syntax [23].

There are many supporting libraries to extend and to get the easier and more powerful implementation of the system. The proposed detection system used some python libraries, especially scikit_learn.

Scikit-learn provides a few functions to split datasets into multiple subsets in various ways. The simplest function is train_test_split, which does pretty much the same thing as the function split_train_test defined earlier, with a couple of additional features. First there is a

random_state parameter that allows you to set the random generator seed as explained previously, and second you can pass it multiple datasets with an identical number of rows, and it will split them on the same indices [24].

This library can also support many machine learning algorithms including CART algorithm. It uses an optimized version of the CART algorithm; however, the scikit-learn implementation does not support categorical variables for now. The selected dataset have numerical data, the output class is assigned by 0 for normal traffic and the positive number of attack traffic. Therefore, it can support to perform the evaluation of the proposed system.

IV. EXPERIMENTS

The experiments are done by using python language and IoT botnet attacks dataset, N-BaIoT.

A. Dataset

The selected dataset, called N-BaIoT [8], was used for the detection system. This dataset has 115 features which are including output class. The output class one benign class and 10 types of attacks class. The traffic data were collected by running two botnet attacks, Mirai and BASHLITE. The following features are included in N-BaIoT dataset [8].

- Stream aggregation
  - H: Stats summarizing the recent traffic from this packet's host (IP)
  - HH: Stats summarizing the recent traffic going from this packet's host (IP) to the packet's destination host.
  - HpHp: Stats summarizing the recent traffic going from this packet's host+port (IP) to the packet's destination host+port. Example 192.168.4.2:1242 -> 192.168.4.12:80
  - HH_jit: Stats summarizing the jitter of the traffic going from this packet's host (IP) to the packet's destination host.
- Time-frame (The decay factor Lambda used in the damped window)
  - How much recent history of the stream is captures in these statistics
  - L5, L3, L1, ...
- The statistics extracted from the packet stream:
  - weight: The weight of the stream (can be viewed as the number of items observed in recent history)
  - mean: ...
  - std: ...
  - radius: The root squared sum of the two streams' variances
  - magnitude: The root squared sum of the two streams' means
  - cov: an approximated covariance between two streams
  - pcc: an approximated covariance between two streams

Although there is 9 kinds of IoT devices records in this dataset, Home_XCS7_1002_WHT_Security_Camera is selected for this analysis because this device is a popular device and it is not expensive. It allows panoramic viewing (355 degrees) and tilt (12 degrees) for optimal viewing coverage two-way talk feature which enables to user to talk back to anyone in the room. It is controlled by the simple home app.

This dataset has 849,255 traffic records and detail description is shown in Table I. The 34% (292,149 records) of the dataset is randomly selected for testing by the scikit-learn library and the remaining part (557,106 records) are for training.

TABLE I. TRAFFIC RECORDS DESCRIPTION FOR SELECTED DATASET

Malware/Normal	Class	Number of Records
Normal	benign (0)	42,784
Mirai	ack (1)	111,480
	scan (2)	45,930
	syn (3)	125,715
	udp (4)	151,879
	udpplain (5)	78,244
Gafgyt	combo (6)	54,283
	junk (7)	18,579
	scan (8)	27,825
	tcp (9)	88,816
	udp (10)	103,720

B. Evaluation Results

The experiment results are based on the implemented program which is created by python language for the CART algorithm. The detection approach is separately done on each kind of attack classes. The 34% of the datasets are used as testing data and the remaining parts are used as training data. According to the results, the average accuracy is more than 99% on the implementation of the detection system.

The training and testing time is shown in Table II. The processing was done on 64 bit Windows 10 in PC (2.8 GHz CPU & 16 GB memory). The total detection (testing) time is 0.15 seconds for 292,147 traffics (records). The processing times are expressed in seconds. The confusion matrix is shown in Table III. These results show that there is no false positive alarm by using the CART algorithm for the detection of botnet attacks on IoT.

TABLE II. PROCESSING TIME (SECONDS)

Process	Time (seconds)
Training	76.80
Testing	0.15

V. CONCLUSION

The IoT devices are very rapidly developed in recent year. On the other side, the attackers are more targeting on these devices. They made the botnet attacks and the infected devices become bots which are controlled by them. After that, they made a serious attack on the targeted systems and devices. Therefore, it is needed to implement an effective detection system for these devices. However, these devices have very limited resources constraint problem. Therefore, the detection model must be lightweight. The proposed detection architecture is implemented by a simple decision tree algorithm. According to the accuracy results, the implemented system using machine learning is very useful for detecting the cyber attack on the smart environment. In the future work, the detection architecture with other learning algorithms will be investigated for detecting cyber attacks on IoT environment.

TABLE III. CONFUSION MATRIX FOR ECACH ATTACKS AND BENIGH DETECTION

Benign/ Attacks	benign	Attack 1	Attack 2	Attack 3	Attack 4	Attack 5	Attack 6	Attack 7	Attack 8	Attack 9	Attack 10
benign	14480	0	1	0	0	0	1	0	2	0	1
Attack 1	0	37723	0	0	0	1	0	0	0	0	0
Attack 2	1	0	15651	0	0	0	1	1	0	0	1
Attack 3	0	0	0	42575	0	0	0	0	0	0	0
Attack 4	0	0	0	0	51845	0	0	0	0	0	0
Attack 5	0	1	0	0	0	26467	0	0	0	1	0
Attack 6	0	0	0	0	0	0	18496	1	0	0	1
Attack 7	0	0	1	0	0	0	1	9685	1	0	0
Attack 8	0	0	0	0	0	0	0	0	9573	0	0
Attack 9	0	0	0	0	0	0	1	0	1	29998	3
Attack 10	0	0	0	0	0	0	0	1	1	0	35632

REFERENCES

[1] Cisco, "Cisco Visual Networking Index (VNI)," Glob. Forecast Update, pp. 1–35, 2017.

[2] Symantec, "Internet Security Threat Report," 2017.

[3] V. K. Mikhail Kuzin, Yaroslav Shmelev, "New trends in the world of IoT threats - Securelist," Kaspersky Lab. 2018.

[4] Snort, "Dissecting Snort," Network, p. 26.

[5] S. A. R. Shah and B. Issac, "Performance comparison of intrusion detection systems and application of machine learning to Snort system," Futur. Gener. Comput. Syst., vol. 80, no. November 2017, pp. 157–170, 2018.

[6] S. D. Bay, D. Kibler, M. J. Pazzani, and P. Smyth, "The UCI KDD archive of large data sets for data mining research and experimentation," ACM SIGKDD Explor. Newsl., vol. 2, no. 2, pp. 81–85, 2000.

[7] L. Dhanabal and D. S. P. Shantharajah, "A Study On NSL-KDD Dataset For Intrusion Detection System Based On Classification Algorithms," Int. J. Adv. Res. Comput. Commun. Eng., vol. 4, no. 6, pp. 446–452, 2015.

[8] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection," no. February, pp. 18–21, 2018.

[9] P. Aggarwal and S. K. Sharma, "An empirical comparison of classifiers to analyze intrusion detection," Int. Conf. Adv. Comput. Commun. Technol. ACCT, vol. 2015-April, pp. 446–450, 2015.

[10] S. Samdani and S. Shukla, "A novel technique for converting nominal attributes to numeric attributes for intrusion detection," 8th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2017, no. 1, pp. 1–5, 2017.

[11] P. Kushwaha, H. Buckchash, and ..., "Anomaly based intrusion detection using filter based feature selection on KDD-CUP 99," Reg. 10 Conf. …, pp. 839–844, 2017.

[12] P. Kushwaha, H. Buckchash, and R. Balasubramanin, "Anomaly based intrusion detection using filter based feature selection on KDD-CUP 99," Proc. 2017 IEEE Reg. 10 Conf. (TENCON), Malaysia, pp. 839–844, 2017.

[13] H. Haddad Pajouh, R. Javidan, R. Khayami, D. Ali, and K.-K. R. Choo, "A Two-layer Dimension Reduction and Two-tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks," IEEE Trans. Emerg. Top. Comput., vol. 6750, no. c, pp. 1–1, 2016.

[14] B. Subba, S. Biswas, and S. Karmakar, "A Neural Network based system for Intrusion Detection and attack classification," 2016 Twenty Second Natl. Conf. Commun., pp. 1–6, 2016.

[15] C. Cervantes, D. Poplade, M. Nogueira, and A. Santos, "Detection of sinkhole attacks for supporting secure routing on 6LoWPAN for Internet of Things," Proc. 2015 IFIP/IEEE Int. Symp. Integr. Netw. Manag. IM 2015, pp. 606–611, 2015.

[16] Z. Guo, I. G. Harris, Y. Jiang, and L. F. Tsaur, "An efficient approach to prevent battery exhaustion attack on BLE-based mesh networks," 2017 Int. Conf. Comput. Netw. Commun. ICNC 2017, pp. 1–5, 2017.

[17] A. Ganesan, P. Parameshwarappa, A. Peshave, Z. Chen, and T. Oates, "Extending Signature-based Intrusion Detection Systems WithBayesian Abductive Reasoning," no. December, 2019.

[18] R. Kumar and D. Sharma, "HyINT: Signature-Anomaly Intrusion Detection System," 2018 9th Int. Conf. Comput. Commun. Netw. Technol., pp. 1–7, 2018.

[19] S. A. R. Shah and B. Issac, "Performance comparison of intrusion detection systems and application of machine learning to Snort system," Futur. Gener. Comput. Syst., vol. 80, no. March, pp. 157–170, 2018.

[20] M. Domb, E. Bonchek-Dokow, and G. Leshem, "Lightweight adaptive Random-Forest for IoT rule generation and execution," J. Inf. Secur. Appl., vol. 34, pp. 218–224, 2017.

[21] M. Alhanahnah, Q. Lin, Q. Yan, N. Zhang, and Z. Chen, "Efficient signature generation for classifying cross-architecture IoT malware," 2018 IEEE Conf. Commun. Netw. Secur. CNS 2018, 2018.

[22] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "Classification and regression trees," Classif. Regres. Trees, vol. 1, no. February, pp. 1–358, 2017.

[23] L. H. Richard, "Learning to Program with Python," John Wiley Sons Inc., p. 310, 2013.

[24] A. Géron, Hands-on machine learning with scikit-learn & tensorflow. 2017.