RADTEC: Re-authentication of IoT Devices with Machine Learning

Kaustubh Gupta and Nirnimesh Ghose
School of Computing
University of Nebraska–Lincoln
kgupta97@huskers.unl.edu and nghose@unl.edu

Boyang Wang
Electrical Engineering and Computer Science
University of Cincinnati
boyang.wang@uc.edu

***Abstract*—The use of Internet of Things (IoT) devices is higher than ever and is growing rapidly. Many IoT devices are manufactured by home appliance manufacturers where security and privacy is not the foremost concern. There does not exist a strict authentication method that verifies the identity of the device. This allows any rogue IoT device to authenticate and spoof various IoT device activities using compromised credentials. This paper addresses the issue by introducing a novel method for re- and continuous authentication utilizing a device-type classification as a new identity paradigm. We present RADTEC: a protocol for authenticating a device in a network by leveraging machine learning to classify the type of an IoT device attempting to connect to the network with an accuracy of over 95% in less than 0.65 milliseconds. We investigate multiple machine learning classifiers to infer the types of IoT devices and use them to develop a stricter and more efficient method for authentication.**

***Index Terms*—IoT security, authentication, device type classification, machine learning, cross layer analysis.**

I. Introduction

The security of Internet of Things (IoTs) devices is prone to two significant vulnerabilities; first, the insecurities and vulnerabilities in the firmware and second, the secrets utilized to bootstrap security are prone to compromise. The Common Vulnerabilities and Exposures (CVE) database of vulnerabilities alone consists of over 1000 records related to the keyword "IoT" depicting the vulnerabilities of firmware [1]. Furthermore, the compromised secrets can be utilized by an unauthorized party to inject/modify sensitive data/actuation [2]–[5]. Therefore, both vulnerabilities compromise the security of the whole network. One of the available ways to address this issue is to use a policy-based access control to prevent insecure devices from taking control of the home network [6]. However, the state-of-the-art requires manual configuration of the policies and is not capable of automatically distinguishing device capabilities to enforce the policy.

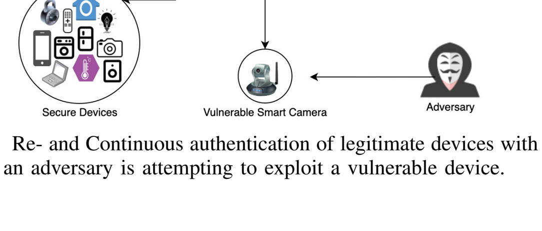

Fig. 1. Re- and Continuous authentication of legitimate devices with the hub, where an adversary is attempting to exploit a vulnerable device.

Previous work includes several attempts to improve the authentication process of an IoT device with a network [7]–[11]. Approaches include authentication based on the proximity of the IoT device [7], [10], while others provide methods of authentication through machine learning by predicting if a device is IoT or not IoT, or by predicting the class of an IoT device such as cameras, hubs, electronics [8], [9], [11]. However, none of the previous state-of-the-art solutions such as proximity-based solutions provide an extensive protocol or methods that are completely independent of any interaction by the user after the initial authentication or machine learning-based solutions where the protocol is independent of vulnerable network characteristics such as the MAC address and a protocol that identifies the type of IoT device with high accuracy and provides a complete and efficient solution that can be used in the real-world. To advance on the previous work done to improve the authentication of an IoT device in a network, the goal of this paper is to develop a protocol that introduces the following key-important features that have not yet been addressed in any of the state-of-the-art solutions:

- Existing works do not address credential compromise [7], [10]. The effect of the compromise can be reduced by introducing an additional hard-to-forge modality for identity verification. We achieve this by using machine learning-based device type classification as a novel modality of identity.
- Several existing machine learning models are only be able to classify a device as IoT or not IoT but not classify the kind of IoT device with high accuracy [8], [11]. We perform device classifications by using traffic fingerprints.
- The machine learning model sometimes cannot make classifications with high accuracy. We address this problem by iterative classification using backup machine learning methods.
- With large amount of data, a machine learning model can use high memory and time, limiting its application for authentication. We increase the efficiency of classification by selecting important quantifiable data.

We propose to tackle the problem, where an adversary can compromise a vulnerable device to control other devices in the network. We introduce a new paradigm of identity for authentication as device type. Such that if an adversary is able to compromise a weak vulnerable device with sensing capabilities, it should not be allowed to mimic a more powerful device such as a home assistance device. Thus, to tackle this problem, we propose using the existing deployment of the device type policy [6] with an ML-based device type classification to limit network access according to device capabilities. We propose a novel technique to perform device type classification and re- and continuous authentication of the devices accordingly. RADTEC - **R**e- and continuous **A**uthentication based on **D**evice **T**yp**E** **C**lassification utilizes cross-layer (network, data link, transport, and application) data to perform classification into six categ ories: Home Assistant, Smart Camera, Smart Electrical and Lighting, Smart Sensor, and Non-IoT devices. The re- and continuous authentication is based on the credentials presented by a device matching the device type in the database. This prevents any adversary from compromising the preloaded secret of a monitoring device and being able to actuate devices in the network.

II. Models and Preliminaries

A. System Model

The system model identified for this work is similar to a network containing IoT devices. The main components of the system are shown in Fig. 2, which are:

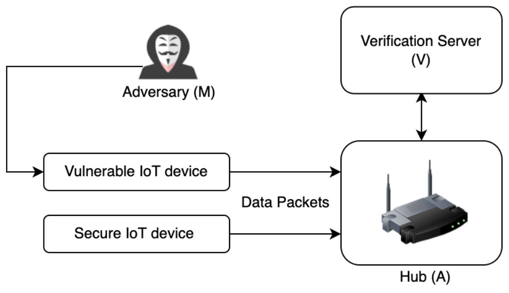

Fig. 2. System Model

Legitimate devices (D): The legitimate devices have already established trust with the network using any existing technique [12]–[14]. There is no limitation to the security requirement and capabilities of the devices.

Hub (A): The hub is responsible for serving legitimate devices. The hub also performs initial trust establishment and verification of existing credentials. The hub provides the connection between the devices and the Internet and is able to see the headers of various layers. The hub is assumed to be physically and cryptographically secure.

Verification Server (V): The verification server is responsible for performing device type classifications based on the traffic pattern. The verification server is accessed by the hub as a cloud service. The hub collects the traffic pattern and transmits it to the verification server to receive the classification. A and V are assumed to have a trusted communication channel.

B. Adversary Model

The adversary (M) is capable of compromising any of the legitimate devices by any method such as but not limited to exploiting the firmware vulnerabilities, or database compromise of pre-shared secrets [15], [16]. The adversary can utilize compromised knowledge to hijack a vulnerable device in the network as an attempt to,

- Poison the network by injecting malicious packets,
- Capture network traffic to extract sensitive data,
- Actuate a device to perform activities of some other type of device.

We assume that the adversary has no prior knowledge of the traffic pattern of any compromised device. This is a reasonable assumption because the adversary, when learning the compromised secrets, does not have access to the legitimate device to capture and perform traffic pattern analysis.

C. Security Requirement

The security requirement of RADTEC is to authenticate devices based on device type classification. The hub is responsible for the verification of the credentials and for the comparison of claimed and observed device types based on the traffic pattern. The hub and the verification server can be assumed to be a single entity as a *secured gateway*. The secured gateway performs 1) initial trust establishment, 2) policy-based network access, and 3) re- and continuous authentication of devices.

III. Re-Authentication based on Device Type Classification

In this section, we present RADTEC - **R**e- and continuous **A**uthentication based on **D**evice **T**yp**E** **C**lassification. The main idea is to first perform device type classification based on traffic pattern. Then utilizing the device type to perform verification during the authentication process.

A. Device Fingerprint Generation

The traffic from the legitimate device (D) is collected by the hub (A). The hub utilizes the quantifiable data in the headers of different layers to collect the device fingerprint. We chose to utilize the header as they are not encrypted because A does not have access to the keys shared between D and cloud services. For generating the fingerprint, we propose using the n number of packets $\{p_D(1), p_D(2), \ldots, p_D(n)\}$ for each device D. For our work, we utilize the data collected by [17]. Initially, we extracted 19 characteristics from each packet, which we define as the feature $f(i,j)$. However, we chose to consider only the important features as removing unnecessary features would improve efficiency in a real world scenario to reduce the time and memory required for model training and classification. We use a random forest search classifier to extract feature importance scores from the 19 features shown in Fig.3. After calculating the importance scores for the features, we set a threshold of 0.05 and eliminated all features except seven.

Now, for a packet $p_D(i)$, we have seven features fingerprint:

$$\mathcal{F}_D = \begin{Bmatrix} f_D(1,1) & f_D(2,1) & \ldots & f_D(n,1) \\ f_D(1,2) & f_D(2,2) & \ldots & f_D(n,2) \\ f_D(1,3) & f_D(2,3) & \ldots & f_D(n,3) \\ \vdots & \vdots & \ddots & \vdots \\ f_D(1,7) & f_D(2,7) & \ldots & f_D(n,7) \end{Bmatrix}. \tag{1}$$

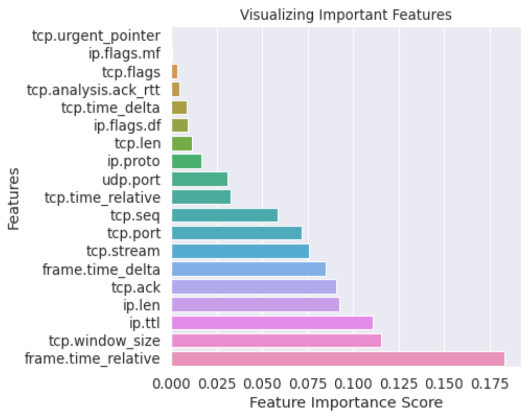

Fig. 3. Feature Importance Scores

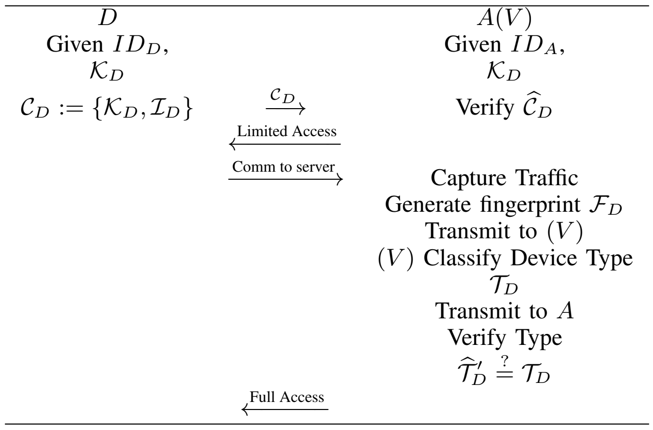
D
Given ID_D,
$\mathcal{K}_D$
$\mathcal{C}_D := \{\mathcal{K}_D, \mathcal{I}_D\}$
$\xrightarrow{\mathcal{C}_D}$
Limited Access
Comm to server
$A(V)$
Given ID_A,
$\mathcal{K}_D$
Verify $\widehat{\mathcal{C}}_D$
Capture Traffic
Generate fingerprint $\mathcal{F}_D$
Transmit to (V)
(V) Classify Device Type
$\mathcal{T}_D$
Transmit to A
Verify Type
$\widehat{\mathcal{T}}'_D \overset{?}{=} \mathcal{T}_D$
Full Access

Fig. 5. Device Authentication using RADTEC protocol.

B. Device Type Classification

We perform classification of devices into seven different types: *smart camera, smart sensor, smart home assistant, smart electrical, smart speaker, and non-IoT*, as shown in Fig 4. The intuition behind is to classify devices as information gathering and actuating devices. This allows for efficient policy implementation, preventing weaker information gathering devices from actuating. We diversify information gathering devices into cameras and sensors as they have different levels of privacy invasion. The camera gives more information about user privacy as compared to the sensors.

We improve the efficiency and accuracy of the classification process by implementing a threshold-based iterative classification technique. **(1)** V selects the initial classifier C_x corresponding to the hub A. **(2)** V computes the type $\mathcal{T}_D(i, x)$ and accuracy $a(i, x)$. **(3)** If the accuracy $a(i, x) \geq \tau$, V securely transmits $\mathcal{T}_D(i, x)$ to A. Otherwise, V sorts $\mathcal{T}_D(i, x)$ according to accuracy $a(i, x)$. **(4)** V selects three types with the highest accuracy $\mathcal{T}_D(i, x)$, $\mathcal{T}_D(j, x)$, and $\mathcal{T}_D(k, x)$. **(5)** V evaluates the fingerprint $\mathcal{F}_D$ using the three classifiers C_i, C_j, and C_k corresponding to the types with the highest accuracy. If any of the accuracy of the classifiers $a(y, y) \geq \tau \ \forall \ y = \{i, j, k\}$, V securely transmits the corresponding type $\mathcal{T}_D(y, y)$ to A. Otherwise, repeat the steps (4) and (5).

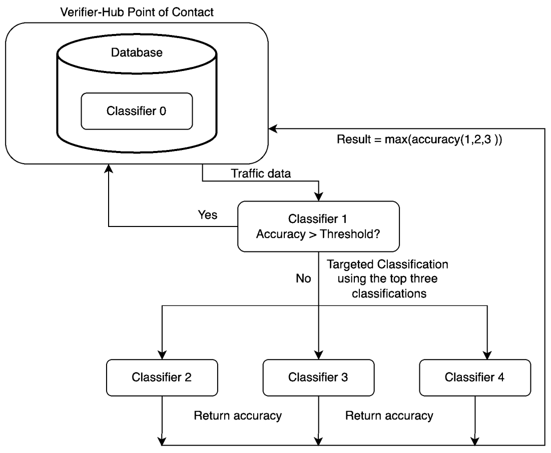

Fig. 4. Verifier

C. RADTEC: The Protocol

RADTEC utilizes the cryptographic credentials and observed device type for authentication of the device, as shown in Fig. 5. First, the hub verifies the validity of the credentials and places the device in a limited network access. During the limited network access period, D is allowed to communicate to the Internet but not to any other entities on the network [18]. This lets the IoT device communicate with the online service. During this communication, A captures the traffic transmitted and received by D. Then A transmits $\mathcal{F}_D$ generated from the captured traffic to the verification server. V computes the device type $\mathcal{T}_D(i, x)$ and returns to A. The hub compares the observed device type with the type saved in the database. If verification is successful, D is given full access to the network according to the deployed policy; otherwise, the device is disconnected and the credentials are marked as compromised.

D. Security Analysis

Now, we analyze RADTEC against the security requirements defined in Section II-C. The adversary (M) can compromise a device and obtain the credential and attempt to connect to A. However, if the adversary is unaware of the device type corresponding to compromised credentials, the adversary will be unable to mimic the traffic pattern and fail the device type verification process. The user will be notified of the compromise and the legitimate device will have to complete the manual initial trust establishment.

RADTEC is capable of identifying whether a device is known or unknown through the use of above mentioned classification techniques. The protocol addresses scenarios in which an adversary can either: 1) exploit a vulnerable device to inject malicious packets and therefore, poison the network [19], 2) use a vulnerable IoT device to extract sensitive data even if the network packets are encrypted [20], or 3) compromise a vulnerable IoT device and actuate the activities that would typically be performed by a different type of device [21].

We detect such attacks by using the classifiers stored in the database as they are trained using the fingerprints of previously authenticated devices. When a machine learning model is asked to make a prediction of a dataset that is similar to the dataset on which it was trained, it gives a highly accurate classification. So, the idea here is that if a device is compromised, the content of the data packets will change, and we can detect those changes by using the classifiers in the database, since the fingerprint will now not be similar to the one used to train the model initially.

This is based on the idea that when a machine learning model is trained and tested on the same or at least similar dataset, it should provide predictions with the highest accuracy. That is also the reason why we capture the traffic of a new device, and once it is authenticated, we train a model solely on the fingerprint of the authenticated device and store that model in the database. Finally, once the verifier sends these results back to the hub, the hub will revoke the full network access and only grant the device limited access to the network. It will further remove the credentials of the compromised device from the list of authenticated and the device will need to be reintroduced into the network.

IV. Implementation

In this section, we discuss the selection of the data set and the process for device identification. We also present the implementation techniques used to classify IoT devices.Finally, we discuss the training of the classifiers used to classify the type of IoT devices and the results.

A. Dataset

Traffic between all devices and the access point was acquired from the University of New South Wales [17]. The data collected include more than 28 IoT devices. We chose 15 devices as shown in Table I and obtained relevant information from packet capture files by extracting important features using *.tshark* into comma separated value files (*.csv*). After capturing the "n" packets from the *.pcap* file and the "f" features using *.tshark*, we built the fingerprint matrix $\mathcal{F}_D$.

TABLE I
List of IoT devices and their classes

Device Name	Category	Class
Amazon Echo	Smart Home Assistant	1
Netatmo Welcome	Smart Camera	2
Samsung SmartCam	Smart Camera	2
Dropcam	Smart Camera	2
Insteon Camera	Smart Camera	2
Belkin Wemo switch	Smart Electrical	3
Light Bulbs LiFX Smart Bulb	Smart Electrical	3
Belkin wemo motion sensor	Smart Sensor	4
Netatmo weather station	Smart Sensor	4
Withings Smart scale	Smart Sensor	4
Withings Aura smart sleep sensor	Smart Sensor	4
Triby Speaker	Smart Speaker	5
Samsung Galaxy Tab	Non-Iot	0
Laptop	Non-Iot	0
iPhone	Non-Iot	0

B. Setup

Algorithm selection: Based on previous work and approaches [7], [8], [22], [23], we decided to use the following five classifiers: Random Forest Classifier (RFC) [24], K-Nearest Neighbors Classifier (KNN) [25], Support Vector Machine Classifier (SVM) [26], Gradient Boosting Classifier (GBC) [27], and Gaussian Naive Bayes Classifier (NAB) [28].

Data Pre-processing: For better classification results the data was preprocessed by using following techniques:

Data Cleaning and Splitting: The *.pcap* files had several data points and characteristics that were not relevant. Therefore, once the *.pcap* files were converted to *.csv* files, we removed all non-quantifiable data from the packets. This included all outgoing traffic from the access point since it does not require classification and would only bias the predictions made by the classifier due to the large amounts of non-quantifiable data present in the packets. The empty columns representing empty values for features were also removed. Finally, the entire dataset was labeled in different classes for training, as shown in Table I. The data in the *.csv* was loaded into a data frame and the labels were separated out of the data frame splitting the data into *features* and *labels*. Then both features and labels were randomly split into train and test data sets in a ratio of 80% and 20%, respectively.

Standardizing Features: Standardizing features is a requirement for many classifiers to achieve high accuracy. We used the standard sklearn scale method to standardize the features in our dataset, which subtracts the mean from the values and then scales it to unit variance,

$$z = \frac{v - m}{s} \tag{2}$$

where, v is the value of the dataset, m is the mean of training samples and s is the standard deviation of the training samples.

Numerical Imputation: We use numerical imputation to assign a value to missing values in any feature. It is better than removing the entire packet since that would effect the amount of data needed to make accurate classifications. Missing values are imputed to the median values of each individual feature. This process is done by utilizing the *fill_na* method provided by the pandas data analysis tool [29].

C. Training and Testing

Training: After collecting and pre-processing the dataset, 80% of it was used to train each of these five classifiers.The training process was performed using the *fit* method provided by *sklearn*, which fits the model onto the data to later provide predictions [30]. The time taken for training was recorded and is shown in Table II.

Testing: The labels for the test data were predicted by the previously trained models, and the predicted labels were compared to the actual labels to calculate the accuracy of the classifications provided by each model. This was done by the use of *accuracy_score* function provided by the *sklearn.metrics* library [31]. Finally, the time required to train each model was calculated and the average time is shown in Table II.

TABLE II
Time required for training and test each model.

Model	Train Time	Test Time
Gaussian Naive Bayes Classifier	50 ms	0.23 ms
Random Forest Classifier	18.17 sec	0.63 ms
N-Nearest Neighbors Classifier	0.34 sec	0.92 ms
Support Vector Machine Classifier	110 min	3.67 ms
Gradient Boosting Classifier	2.9 min	6.89 ms

D. Results

The metrics used to analyze each model performance were the F1 score, the Receiver Operator Characteristic - Area Under the Curve (ROC-AUC) Score, and the accuracy score.

F1 Score: The F1 score is an evaluation metric used to determine the performance of a machine learning classifier and is defined as the harmonic mean of recall and precision. It gives a better insight about the classification made by each device type classifier as it not only calculates the number of misclassifications made by the different models but helps identify the types of mislassifications made.

Precision, also known as the positive predicted value, is the ratio between the number of correct true positives and the total number of instances predicted to be positive and given by:

$$Precision = \frac{T_P}{T_p + F_p}, \tag{3}$$

where T_p is number of true positives, and F_p is the number of false positives.

Recall, also known as sensitivity, is the ratio between the correct true positives and the total sum of the number of false negatives and true positives, and is given by:

$$Recall = \frac{T_p}{T_p + F_n} \tag{4}$$

where T_p is number of true positives, and F_n is the number of false negatives.

The value of the F1 score can be in the range of 0 and 1, where 1 is the highest score a model can achieve. The formula for the F1 score is given by [32]:

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{5}$$

The F1 score for each class within each model is plotted in the Fig. 6.The highest average F1 score for all classes was provided by the RFC. Further, the model can also perfectly differentiate between an IoT and a non-IoT device.

ROC-AUC Score: The ROC AUC score is the area under the receiver operating characteristics (ROC) curve that is used to evaluate the performance of a machine learning model. For our study, we show the ROC AUC scores in Table III. The results attests our findings that RFC has the best performance.

TABLE III
ROC-AUC SCORE.

Model	ROC AUC Score
Random Forest Classifier	0.9954
K-Nearest Neighbors	0.9313
Support Vector Machine Classifier	0.9642
Gradient Boost Classifier	0.9861
Gaussian Naive Bayes	0.9187

Accuracy Score: The accuracy score is an evaluation metric used for machine learning models to measure their performance by determining the ratio between the number of correct predictions made by the classifier and the total number of predictions to be made (6).

$$Accuracy\ Score = \frac{N_c}{N_t} \tag{6}$$

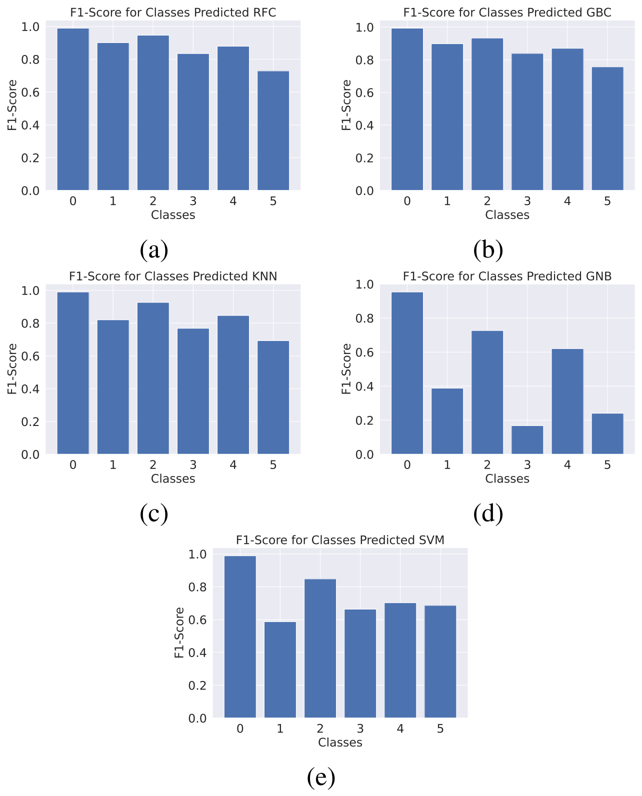

Fig. 6. F1 scores for (a) RFC, (b) GBC, (c) KNN, (d) GNB, and (e) SVM.

where N_c is the number of correct predictions, and N_t is the number of total predictions.

The function to calculate the accuracy of our machine learning models used to perform multi-class classification was provided by the *sklearn.metrics* library [31]. After making the predictions, we established that the Random Forest Classifier (RFC) was the most accurate model to make the predictions with an accuracy of 95.2%. The second most accurate classifier was GBC with an accuracy of 94.8%, then the KNN with accuracy 93.3%, SVM with accuracy of 88.3%, and finally NAB with accuracy of 76.8%, shown in Fig. 7.

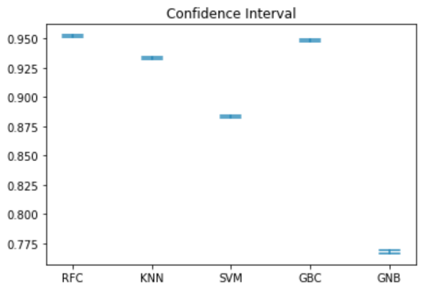

Fig. 7. Accuracy with confidence intervals for different algorithms.

Finally, for our work, we utilized the data collected by [17]. The authors of the data present an IoT device type classification method that uses multiple classifiers trained on different types of quantifiable and textual data. They present a combined classifier with an accuracy over 99%. However, even though they provide a highly accurate model for classifying the different types of IoT devices, they do not provide a formal authentication protocol that can utilize the proposed classification techniques. Furthermore, we believe that our approach is more efficient, as it relies only on quantifiable data to provide a classification.

V. Conclusion

We addressed the problem where a rogue IoT device can spoof activities including sensing, actuation, controlling, etc. using compromised credentials. We presented a novel method for continuous and re-authentication utilizing a device type classification as a new identity paradigm. We presented an authentication protocol: RADTEC leveraging Machine Learning (ML) to classify the type of IoT device attempting to connect to the network with an accuracy of over 95% within 0.65 ms. We compared different types of machine learning classifiers to best estimate the types of IoT devices and used them to develop a stricter and efficient method for authentication.

In our future work, we would train classification models for several more types of IoT devices. We would also collect more data for each type of device to eliminate bias in our model, providing more accurate classifications for all the devices considered. In addition to the machine learning models used in this thesis, we would perform device type level classification using several other types of classifiers and apply model fine-tuning techniques to improve their performance.

Acknowledgement

The authors would like to thank the anonymous reviewers for their comments and suggestions. The UC author was partially supported by National Science Foundation (CNS-1947913). Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the author(s) and do not necessarily reflect the views of the NSF.

References

[1] CVE, "IoT search results," https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=IoT, [Online; accessed 20-Mar-2022].

[2] T. Armerding, "The 18 biggest data breaches of the 21st century," https://www.csoonline.com/article/2130877/the-biggest-data-breaches-of-the-21st-century.html, 2018, [Online; accessed 10-Feb-2020].

[3] M. Vanhoef and F. Piessens, "Key reinstallation attacks: Forcing nonce reuse in WPA2," in *Proc. of ACM Conference on Computer and Communications Security (CCS)*. ACM, 2017.

[4] S. Havron, D. Freed, R. Chatterjee, D. McCoy, N. Dell, and T. Ristenpart, "Clinical computer security for victims of intimate partner violence," in *Proc. of USENIX Security Symposium*, 2019, pp. 105–122.

[5] D. Freed, J. Palmer, D. Minchala, K. Levy, T. Ristenpart, and N. Dell, ""A Stalker's Paradise": How Intimate Partner Abusers Exploit Technology," in *Proc. of CHI Conference on Human Factors in Computing Systems*. ACM, 2018, p. 667.

[6] C. Meraki, "Applying policies by device type," https://documentation.meraki.com/MR/Group_Policies_and_Block_Lists=/Applying_Policies_by_Device_Type, 2021, [Online; accessed 14-Feb-2021].

[7] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A.-R. Sadeghi, and S. Tarkoma, "IoT sentinel: Automated device-type identification for security enforcement in IoT," in *Proc. of International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 2177–2184.

[8] A. Bremler-Barr, H. Levy, and Z. Yakhini, "IoT or not: Identifying IoT devices in a short time scale," in *Proc. of Network Operations and Management Symposium*. IEEE, 2020, pp. 1–9.

[9] Y. Liu, J. Wang, J. Li, S. Niu, and H. Song, "Machine learning for the detection and identification of internet of things devices: A survey," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 298–320, 2021.

[10] J. Zhang, Z. Wang, Z. Yang, and Q. Zhang, "Proximity based IoT device authentication," in *Proc. of IEEE INFOCOM*. IEEE, 2017, pp. 1–9.

[11] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "DÏoT: A federated self-learning anomaly detection system for IoT," in *Proc. of International conference on distributed computing systems (ICDCS)*. IEEE, 2019, pp. 756–767.

[12] H.-A. Wen, T.-F. Lee, and T. Hwang, "Provably secure three-party password-based authenticated key exchange protocol using weil pairing," *IEEE Proceedings-Communications*, vol. 152, no. 2, pp. 138–143, 2005.

[13] M. Vanhoef and E. Ronen, "Dragonblood: Analyzing the dragonfly handshake of wpa3 and eap-pwd," in *Proc. of IEEE S&P*. IEEE, 2020.

[14] N. Ghose, L. Lazos, and M. Li, "SFIRE: Secret-free in-band trust establishment for COTS wireless devices," in *Proc. of IEEE INFOCOM*, 2018, pp. 1529–1537.

[15] J. Liu and W. Sun, "Smart attacks against intelligent wearables in people-centric internet of things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 44–49, 2016.

[16] S. Pallavi and V. A. Narayanan, "An overview of practical attacks on BLE based IoT devices and their security," in *Proc. of ICACCS*. IEEE, 2019, pp. 694–698.

[17] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Classifying IoT devices in smart environments using network traffic characteristics," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, 2018.

[18] C. Haar and E. Buchmann, "Fane: A firewall appliance for the smart home," in *Proc. of Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, 2019, pp. 449–458.

[19] S.-Y. Hwang and J.-N. Kim, "A malware distribution simulator for the verification of network threat prevention tools," *Sensors*, vol. 21, no. 21, p. 6983, 2021.

[20] A. Acar, H. Fereidooni, T. Abera, A. K. Sikder, M. Miettinen, H. Aksu, M. Conti, A.-R. Sadeghi, and S. Uluagac, "Peek-a-boo: I see your smart home activities, even encrypted!" in *Proc. of ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2020, pp. 207–218.

[21] J. Yun, I.-Y. Ahn, J. Song, and J. Kim, "Implementation of sensing and actuation capabilities for IoT devices using oneM2M platforms," *Sensors*, vol. 19, no. 20, p. 4567, 2019.

[22] M. A. Al-Garadi, A. Mohamed, A. K. Al-Ali, X. Du, I. Ali, and M. Guizani, "A survey of machine and deep learning methods for internet of things (IoT) security," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1646–1685, 2020.

[23] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT security techniques based on machine learning: How do IoT devices use ai to enhance security?" *IEEE Signal Processing Magazine*, vol. 35, no. 5, pp. 41–49, 2018.

[24] Scikit-Learn, "Sklearn.ensemble.randomforestclassifier," https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForest Classifier.html, [Online; accessed 22-Mar-2022].

[25] ——, "Sklearn.ensemble.kneighborsclassifier," https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighbors Classifier.html, [Online; accessed 22-Mar-2022].

[26] ——, "Sklearn.ensemble.svm.svc," https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html, [Online; accessed 22-Mar-2022].

[27] ——, "Sklearn.ensemble.gradientboostingclassifier," https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoosting Classifier.html, [Online; accessed 22-Mar-2022].

[28] ——, "Sklearn.ensemble.naivebayes.gaussiannb," https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html, [Online; accessed 22-Mar-2022].

[29] Pandas, "pandas.dataframe.fillna," https://pandas.pydata.org/docs/reference /api/pandas.DataFrame.fillna.html, [Online; accessed 06-May-2022].

[30] Scikit-Learn, "Developing scikit-learn estimators," https://scikit-learn.org/stable/developers/develop.html, [Online; accessed 24-Mar-2022].

[31] ——, "sklearn.metrics.accuracy_score," https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html, [Online; accessed 24-Mar-2022].

[32] ——, "sklearn.metrics.f1_score," https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html, [Online; accessed 24-Mar-2022].