Spring 4-29-2022

Machine Learning-Based Device Type Classification for IoT Device Re- and Continuous Authentication

Kaustubh Gupta
University of Nebraska-Lincoln, kgupta97@huskers.unl.edu

MACHINE LEARNING-BASED DEVICE TYPE CLASSIFICATION FOR IOT DEVICE RE- AND CONTINUOUS AUTHENTICATION

by

Kaustubh Gupta

A THESIS

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfilment of Requirements

For the Degree of Master of Science

Major: Computer Science

Under the Supervision of Professor Nirnimesh Ghose

Lincoln, Nebraska

May, 2022

MACHINE LEARNING-BASED DEVICE TYPE CLASSIFICATION FOR IOT DEVICE RE- AND CONTINUOUS AUTHENTICATION

Kaustubh Gupta, M.S.

University of Nebraska, 2022

Adviser: Nirnimesh Ghose

Today, the use of Internet of Things (IoT) devices is higher than ever and it is growing rapidly. Many IoT devices are usually manufactured by home appliance manufacturers where security and privacy are not the foremost concern. When an IoT device is connected to a network, currently there does not exist a strict authentication method that verifies the identity of the device, allowing any rogue IoT device to authenticate to an access point. This thesis addresses the issue by introducing methods for continuous and re-authentication of static and dynamic IoT devices, respectively. We introduce mechanisms and protocols for authenticating a device in a network through leveraging Machine Learning (ML) to classify not only if the device is IoT or not but also the type of IoT device attempting to connect to the network with an accuracy of over 95%. Furthermore, we compare different types of machine learning classifiers to best estimate the types of IoT devices and use them to develop a stricter and more efficient method of authentication.

DEDICATION

To my parents for always supporting me and making it possible for me to chase my dreams.

ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude and appreciation to my academic advisor Dr. Nirnimesh Ghose for his advice, support, motivation, and patience throughout the last two years. Through his vast knowledge and experience, he taught me where to explore, how to think while performing research and how to present my work. I am extremely fortunate to have him as my advisor and as my mentor. This thesis would have been impossible without his guidance.

I am grateful to Dr. Byrav Ramamurthy for his support and guidance throughout my time as an undergraduate and then as a graduate student. My career as a graduate student at the University of Nebraska - Lincoln would not have been possible without him.

I would like to thank Dr. Lisong Xu and Dr. Byrav Ramamurthy for serving on my master's committee and for providing insightful and valuable input that helped me make this thesis better.

I am extremely grateful to my mother Kanika Gupta, my father Sandeep Gupta, and my sister Anubhuti Gupta for their love, prayers, support, and guidance throughout my life; you have always been my source of inspiration, confidence, and success. I am also grateful and fortunate to have my partner and best friend Kelbie Schnieder and her parents Patti Schnieder and Kent Schnieder. Your love, care and support give me the strength to overcome any difficulties that I may face.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The Internet of Things (IoTs) stems from the idea of interconnecting most contemporary devices in the network. Many of these devices are wirelessly connected to facilitate the deployment process. Recently, there has been an explosion of embedding wireless capabilities in several devices, which is expected to reach 42 billion devices worldwide by 2025 [17]. Network connected devices include devices such as internet-enabled appliances, medical devices, smart locks, wearables, home monitoring sensors, cameras, industrial sensors and actuators, and many more [3, 19, 13]. These devices collect a large amount of sensitive information about the user's whereabouts, health, behavior, and environment [13]. They are also responsible to perform tasks that are safety-critical, such as safe flying of UAVs, automatically regulating one's heart rate and delivering drugs, controlling entry to one's residence, controlling gas and electric appliances, etc. [13]. For example, a smart garage door provides access to the house premises, a remotely programmed pacemaker controls the electrical pulses applied to the heart [42, 13], and a smart insulin pumps continuously monitor and adjust insulin delivered to diabetic patients [18, 13].

The security of these 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 900 records related to the keyword "IoT" depicting the vulnerabilities of firmware [10]. Furthermore, the compromised secrets can be utilized by an unauthorized party to inject/modify sensitive data [5, 38, 8, 15, 11]. 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 [23]. 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.

1.1 Motivation

Previous work includes several attempts to improve the authentication process of an IoT device with a network. Different approaches include authentication based on the proximity of the IoT device, 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. 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 the previous work done to improve the authentication of an IoT device

in a network, the goal of this thesis is to propose a protocol that introduces the following key-important features that have not yet been addressed in any of the previous state-of-the-art solutions;

- The scenario of credential compromise has not been addressed in previous work. The effect of the compromise can be reduced by minimizing the user interaction after initial password authentication of the IoT device with the access point. One way to achieve this is to automate the IoT authentication process through the use of machine learning.

- Several existing machine learning models do not classify the type of IoT device; therefore, the machine learning models should be able not only to classify a device as IoT or not IoT, but also to classify the type of IoT device that attempts to authenticate with the network with high accuracy.

- If the primary machine learning model cannot classify a device with high accuracy, the protocol should take further steps to provide a classification using backup machine learning methods.

- The proposed protocol should consume the least amount of time and memory for each authentication instance and should not depend on the vulnerable characteristics of the device or the network, such as MAC addresses, which can easily be spoofed.

1.2 Contribution

The proposed technique utilizes cross-layer data including network, data link, transport, and application to perform the device type level classification. The intuition is to fingerprint the IoT device behavior and prevent them from having advanced access

available to non IoT devices such as computers and smartphones. However, this still leaves vulnerable IoT devices as the weak link to the network. Such as in a home network, it would not prevent monitoring IoT devices such as a camera to perform actuation such as opening a garage door. The deployed policy will give the same level of capabilities to all IoT devices.

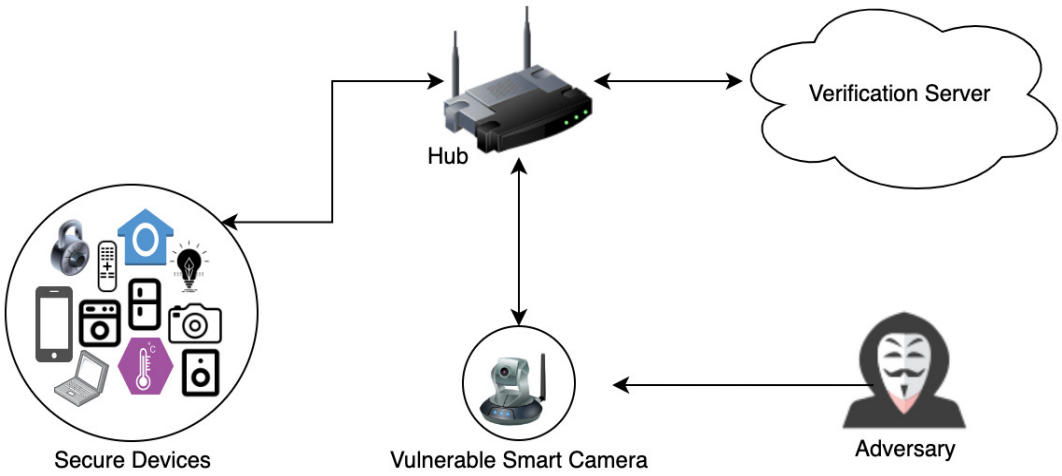

Figure 1.1: Thesis Overview

In this thesis, we address the case where a vulnerable IoT device can be compromised by an adversary, where it can be manipulated to 1. capture network traffic, 2. poison the network by uploading malicious packets and, 3. actuate a device to perform activities of some other type of device, such as allowing a smart camera to operate as a garage door opener, as shown in Fig. 1.1 showing the overview of the setup.

Thus, to tackle this problem, we propose using the existing deployment of the device type policy [23] with an ML-based device type classification to limit network access according to device capabilities. In contrast to existing work, we propose a novel technique to perform device-type classification and re- and continuous authen-

tication of the devices accordingly. RADTEC - **R**e- and continuous **A**uthentication based on **D**evice **T**yp**E** **C**lassification utilizes cross-layer data (network, data link, transport, and application) to perform classification into six categories: 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 match the device type in the database. This prevents any adversary from compromising the preloaded secret of a monitoring device and from being able to actuate devices on the network. The main contribution of our work is as follows:

- We present a device type-based re- and continuous authentication protocol. The protocol is capable of preventing any adversary with a compromised secret from imitating more advanced devices.
- We perform extensive theoretical security analysis to prove the security of the proposed technique against an advanced adversary capable of compromising weak IoT devices.
- We perform extensive experimentation by utilizing the available data [37] for our ML-based device type classification technique to show the performance of various classifiers based on algorithms including Random Forest [32], K-Nearest Neighbors [30], Support Vector Machine [33], Gradient Boosting [29], and Gaussian Naive Bayes [31]. We also show that the Random Forest Classifier [32] is the most efficient in performing accurate classifications.

Chapter 2

Background and Related Work

The problem of developing a strict authentication protocol for IoT devices has recently been tackled, where some researchers have proposed machine learning-based techniques to differentiate several types of IoT devices from non-IoT devices, while others have proposed proximity-based solutions. In this section, we discuss the work done by several researchers in the related area. We then describe the outcomes and limitations of their work.

2.1 Machine Learning Based Authentication

In recent work, the authors have provided a system capable of automatically identifying the type of IoT device through the use of machine learning (ML) and limiting the communication of vulnerable devices to minimize damage inflicted on the network [24]. However, the protocol relies on MAC addresses to identify a new device trying to authenticate with the network, which can be spoofed. Furthermore, the work does not consider the case where a previously authenticated device is compromised or a re-authentication process for every time the device is reintroduced into the network. Bremler *et al.* focus on distinguishing between IoT a Non-IoT (NoT) devices through the use of ML classifiers to assign relevant security policies to the device [7].

However, among several drawbacks, the biggest limitation is demonstrated by their classification technique since their classification model is only capable of classifying a device as IoT or not-IoT. Not addressing this issue would allow the adversary to compromise a vulnerable device to actuate activities that should only be performed by a different kind of device, for example, using a vulnerable smart camera to open a lock or a garage door. In another work, the authors aim to automatically detect suspicious IoT devices in a network through the Random Forest classifier and white list devices that are classified as trustworthy [22]. The drawback of this approach is that if a vulnerable device that has already been whitelisted is compromised by an adversary, it would give the adversary unrestricted access to the network, since the network is not capable of identifying the change in device behavior. In other work, the authors propose IoT security solutions based on ML techniques, including reinforcement learning, unsupervised learning, and supervised learning to improve IoT systems spoofing resistance and detection and to authenticate a device to protect data privacy [41]. Their work attempts to detect an attack through several machine learning techniques; however, each attack is identified through a different machine learning model, which can end up utilizing a large number of resources such as memory and time. Furthermore, in our opinion, if the models are trained to detect an attack by observing a certain pattern, the attack in real-life might differ from what the models are trained to identify which could potentially leave attacks undetected.

A survey related to ML-based classification techniques to detect and identify legitimate and rogue IoT devices to provide security where conventional approaches that use cryptographic protocols cannot be applied [21]. However, the paper demonstrates limitations as it does not present a complete and formal authentication protocol that can incorporate the ML techniques presented in order to protect the network from

certain attacks in real life. An in-depth survey of different machine learning techniques that can be used in the field of IoT to intelligently monitor the security of IoT devices to implement certain security measures such as authentication, network and application security, access control, and encryption is presented in [4]. The work focuses mainly on detecting attacks by monitoring the behavior of the IoT device; however, it does not extend machine learning classification techniques to differentiate between several types of IoT devices.

Finally, for our work, we utilize the data collected by [37]. 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 such as DNS query content, port numbers, cipher suite, etc. Finally, 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.

2.2 Proximity Based Related Work

The state-of-the-art proximity-based solution proposed in [44] is based on several physical activities performed by a user, such as moving a smartphone towards and away from an IoT device and rotating the smartphone to authenticate an IoT device. The work provides a notable contribution in the field of IoT device authentication; however, it requires a significant amount of work to be performed by the user. Furthermore, the authors only address initial authentication and do not discuss the measures

that must be taken if an already authenticated vulnerable device is compromised.

In other work, the authors propose a proximity-based user authentication solution for voice-powered IoT devices [14]. The work presents a voice-based distance estimation technique to authenticate IoT devices using technologies such as Bluetooth, speakers, and microphone. However, the biggest constraint for the proposed method is that it is only applicable to voice-powered IoT devices. Shafagh and Hithnawi propose another proximity-based solution for IoT device authentication *by solely utilizing the wireless communication interface* [35]. Their method attempts to differentiate a legitimate request for authentication from an illegitimate one by the use of ambient radio signals that estimates the proximity of an IoT device. However, their solution presents limitations, since it does not account for a device in close proximity being compromised due to its security vulnerabilities, which could lead an adversary to perform attacks such as actuation, poisoning the network, and capturing network traffic. The techniques used for device identification in [25] only identify if a device has been compromised and do not provide a formal authentication protocol that can utilize the proposed solution to authenticate an IoT device before establishing a connection with the network. Finally, in other work, the authors propose a device identification based on fingerprint recognition of the wireless device chipset [27]. However, their solution is not capable of identify a compromised legitimate device.

Chapter 3

Models and Preliminaries

In this chapter, we first present the system model, which consists of three main components: legitimate devices, a hub, and a verification server. Next, we discuss the adversary model, which discusses several possible ways that an adversary can exploit a vulnerable device to: 1. capture network traffic, 2. poison the network by injecting malicious packets, and 3. actuate a device to perform activities of some other type of device. Furthermore, we present the security requirements of the RADTEC protocol to authenticate devices based on device-type classification, and finally, we give a brief overview of the machine learning techniques and algorithms we employ in this thesis.

3.1 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. 3.1, which are:

Legitimate devices (D): Legitimate devices have already established trust with the network using any existing technique [40, 39, 12]. There is no limitation to the security requirement and capabilities of the devices.

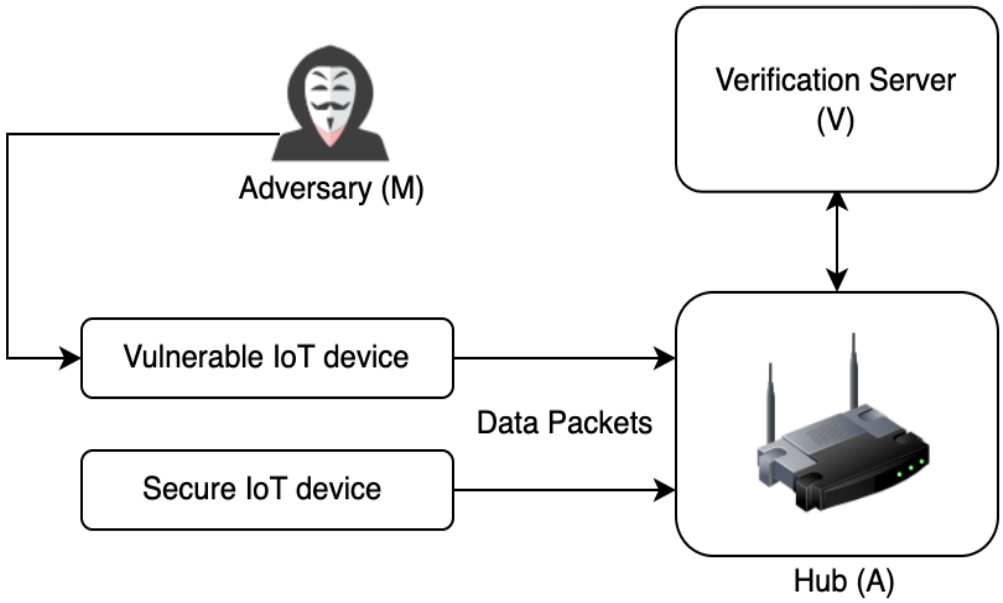

Figure 3.1: System Model

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.

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. This channel can be realized using any contemporary cryptographic technique so that an authenticated encryption AE_K$(\cdot)$ can be implemented [6].

3.2 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 [20, 26]. 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 legitimate 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.

3.3 Security Requirement

The security requirement of RADTEC is to authenticate devices based on the classification of the device type. 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 and authentication of devices.

The first can be achieved using any of the existing methods [40, 39, 12]. The assumption here is that, after the initial user-initiated trust establishment, each device

is assigned independent credentials. This is already present in existing technologies such as WiFi Protected Setup (WPS), where device-specific Pre-Shared Keys (PSK) are assigned for WPA2 or WPA 3 [39]. These keys can be utilized for any subsequent authentication or to build future security properties of integrity verification or confidentiality.

For the second, the network can implement levels of network access based on known vulnerabilities of a device type [24]. To implement these policies on a microlevel, we developed a classification technique that can distinguish between various types of IoT devices. The known vulnerabilities of these device types can be extracted from a vulnerability database such as the Common Vulnerabilities and Exposures (CVE) [10] and utilized to tailor the policies.

Finally, the intuition behind the third is to provide an additional modality during the re- and continuous authentication phase. In addition to the credentials, the behavior of the device should match previously known behavior. The secured gateway saves the traffic fingerprints with the credentials and utilizes them as parameters for during re-authentication. Therefore, an adversarial device now not only has to compromise credentials, but also mimics the known traffic pattern of the compromised device to authenticate with the network.

3.4 Preliminaries - Machine Learning Models

In this section, we describe supervised learning and five different types of supervised machine learning algorithms used to classify the type of IoT device in the network. These algorithms include the Random Forest Classifier, K-Nearest Neighbors, Sup-

port Vector Machine Classifier, Gradient Boost Classifier, and Naive Bayes Classifier.

3.4.1 Supervised Learning (SL)

Supervised Learning is a sub-category of machine learning where the learning of an algorithm is supervised. This essentially means that the algorithm is taught by using examples. As shown in Fig.3.2, the data collected is first labeled into its respective categories by a supervisor. This data is then pre-processed and split into train and test datasets. The algorithm uses training data that consist of labeled input data for training, where it searches for patterns and then correlates each data point with its respective label. Then, for prediction, the unsupervised machine learning algorithms take the unseen test data and attempt to make a determination of its label by using patterns learned during the training process.

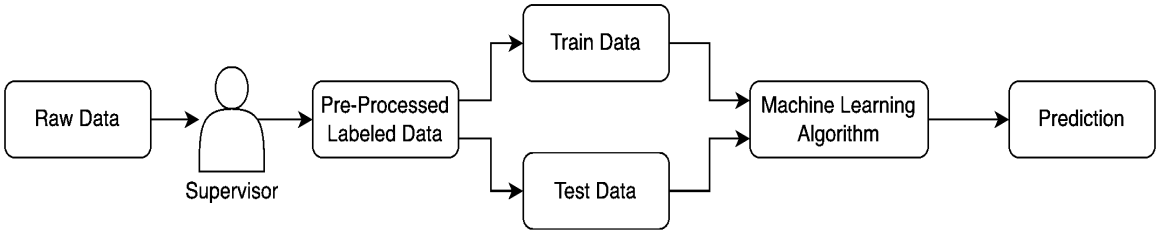

Figure 3.2: Supervised Learning

3.4.2 Random Forest Classifier

The Random Forest Classifier is based on a decision tree like structure at its core, and it can be categorized as an ensemble-based learning method used for making classifications. The algorithm is called ensemble-based because it makes a prediction using an ensemble of large amounts of different and completely uncorrelated decision trees. The final result is based on the predictions made by each individual decision

tree where the class with the majority of votes is the models final prediction.

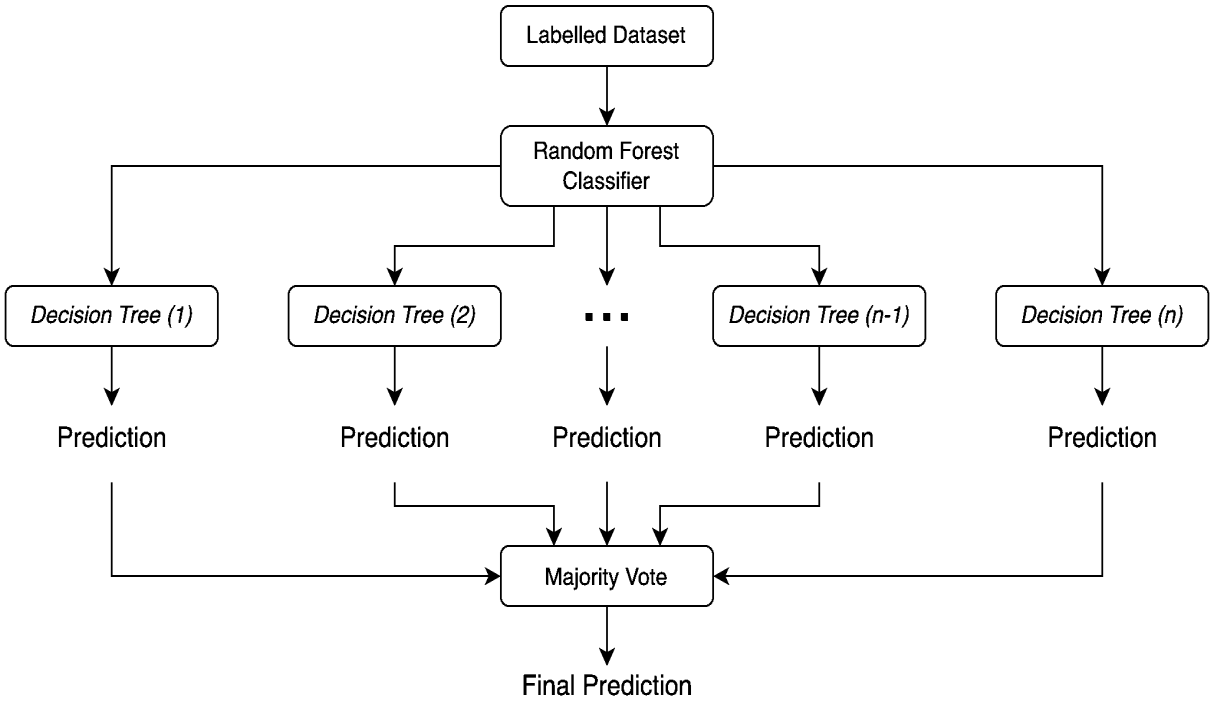

Figure 3.3: Random Forest Classifier

3.4.3 K-Nearest Neighbors Classifier

The K-Nearest Neighbor (KNN) classifier is a supervised machine learning algorithm, mostly used for solving classification problems. The algorithm works by estimating the test data point in a group, based on its nearest "K" number of neighbors. Furthermore, the algorithm does not require any training; instead, it stores the training dataset and considers the training data points as neighbors of each test data point during the classification process. For example, if K=5, the algorithm will look at the five nearest neighbors (from the training data set) of the test data point, and if three out of five neighbors belong to class A and two out of five belong to class B, the final

classification of the test data point will be class A.

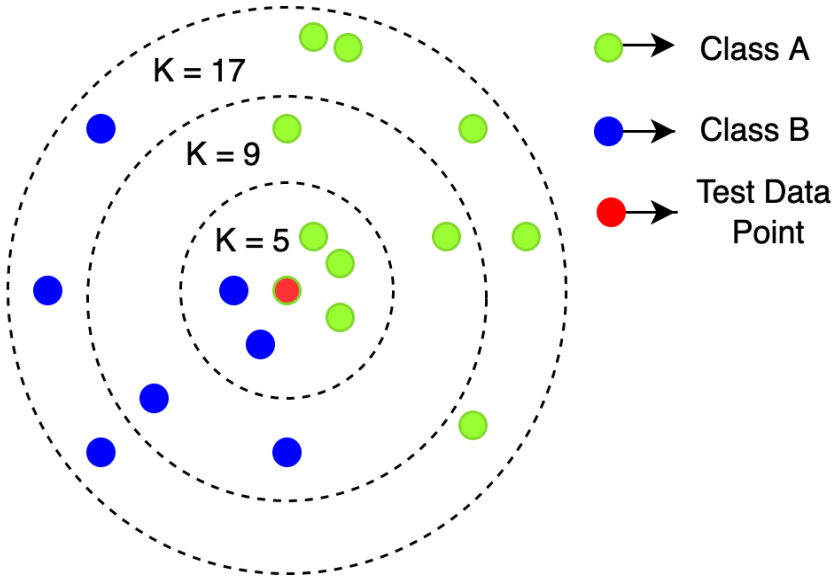

Figure 3.4: K-Nearest Neighbors Classifier

3.4.4 Gradient Boosting Classifier

The Gradient Boosting Classifier is a supervised machine learning algorithm based on the ensemble technique, which means that it utilizes the predictions made by several different weak decision trees to give a strong final prediction. The gradient boosting algorithm uses an additive approach to build the model by typically adding several decision trees sequentially, where in each iteration, the successor tree utilized the results generated by its predecessor tree to reduce error. The processes, also shown in Fig.3.5, effectively reduces the error over several iterations, which helps the algorithm to provide its final predictions.

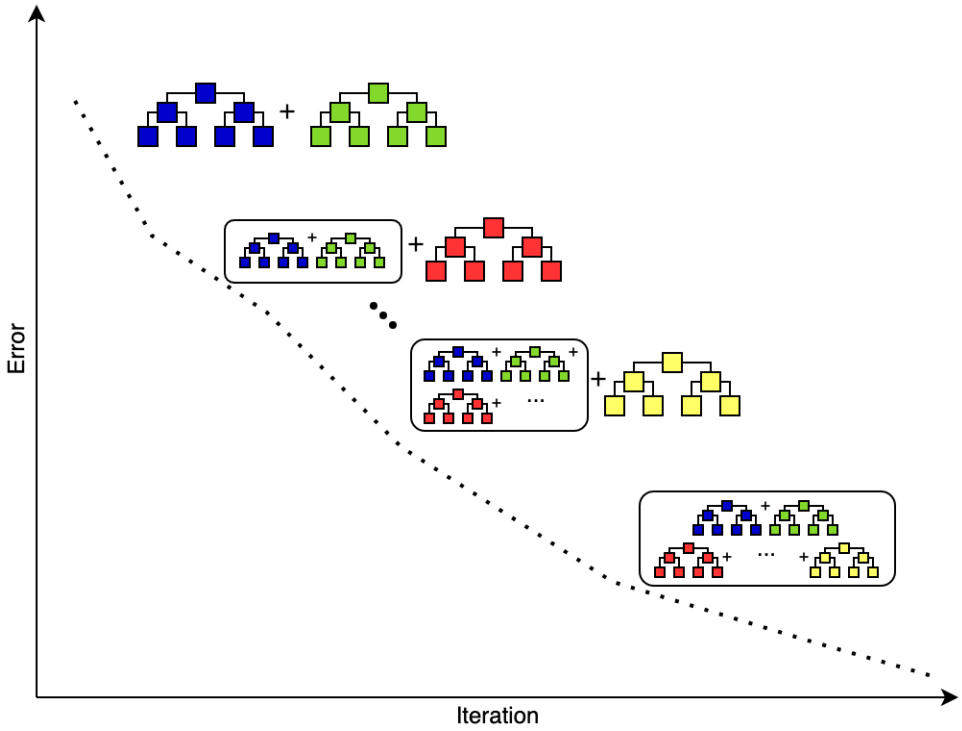

Figure 3.5: Gradient Boosting Classifier

3.4.5 Support Vector Machine Classifier

The Support Vector Machine Classifier is a supervised machine learning algorithm, mostly used for solving classification problems. The algorithm plots all data points with an 'n' number of features in an n-dimensional space, and the coordinate value of each data point is the value of the feature. Finally, classification is performed by finding hyperplanes that differentiates the multiple classes, and if a test data point can be placed within a certain hyperplane, it will share the same class with the data points in its neighborhood.

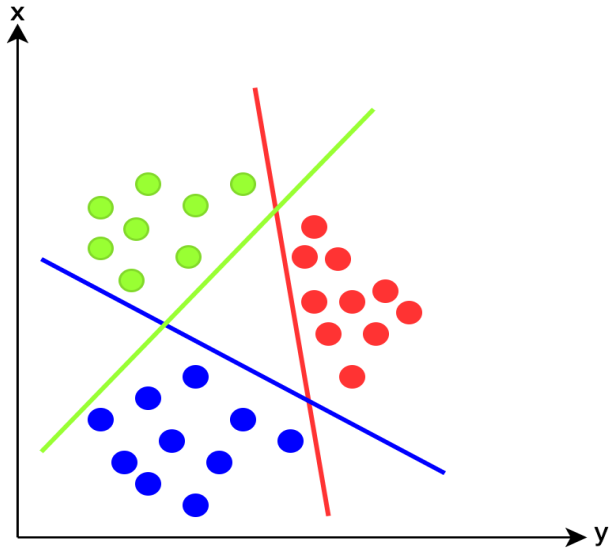

Figure 3.6: Support Vector Machine Classifier

3.4.6 Gaussian Naive Bayes Classifier

The Gaussian Naive Bayes Classifier is often used for classification jobs where the values of all features are continuous and distributed in a Gaussian distribution. The algorithm is called naive because it implies that the presence of any feature is completely independent of the existence of any other feature. It is based on the Bayes theorem (Eq.3.1) which helps define the probability of the occurrence of hypothesis A after the data B, is already given.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{3.1}$$

Chapter 4

Protocol

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 additional verification during the authentication process. Before diving into the protocol, we present the machine learning-based device type classification technique. For device-type classification, we first generate the fingerprint using features embedded in the packet and use them to perform device-type classification.

4.1 Device Fingerprint Generation

The traffic from the legitimate device (D) is collected by the hub (A). The hub utilizes the unique characteristics in the headers of different layers to collect device fingerprint. We chose to utilize the header as they are not encrypted and 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 [37]. 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 by reducing the time required for model training and classification and the required memory. Therefore, we calculate the importance scores for the features by utilizing the results provided by making predictions using a basic random forest classifier. We chose seven out of 19 features for which the importance score is greater than 0.05, as shown in Table 4.1. We first selected this threshold by keeping the top five most important scores and considered more features until the model performance was either unchanged or negatively affected.

Feature	**OSI Model Layer**	**Importance Score**
tcp.port	Transport Layer	0.066480
tcp.stream	Transport Layer	0.094845
frame.time_delta	Physical Layer	0.096504
ip.len	Network Layer	0.099793
ip.ttl	Network Layer	0.102245
tcp.window_size	Transport Layer	0.125575
frame.time_relative	Physical Layer	0.163713

Table 4.1: Features with corresponding importance score.

Now, for a packet $p_D(i)$, from D we have seven fingerprint characteristics such as,

$$\mathcal{F}_D = \begin{Bmatrix} f_D(1,1) & f_D(2,1) & \dots & f_D(n,1) \\ f_D(1,2) & f_D(2,2) & \dots & f_D(n,2) \\ f_D(1,3) & f_D(2,3) & \dots & f_D(n,3) \\ \vdots & \vdots & \ddots & \vdots \\ f_D(1,7) & f_D(2,7) & \dots & f_D(n,7) \end{Bmatrix} \tag{4.1}$$

4.2 Device Type Classification

We perform classification of devices into seven different types: *smart camera, smart sensor, smart home assistant, smart electrical and lighting, smart speaker, and non-IoT*. The intuition behind choosing these types are camera and sensors collect information and home assistants can perform actuation. Thus, this will allow efficient implementation of policies, preventing information gathering devices from actuating. In addition, we diversify information gathering devices into cameras and sensors as they collect data with 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. The collected dataset is first divided into 80% for training and 20% for testing, then the five different models are trained by fitting the training data on each model individually.

1. **Initialization:** After establishing a secure connection with the verifier, the hub A transmits the fingerprint $\text{AE}(\mathcal{F}_D)$ to the verification server V, using the trusted channel.

2. **Initial Classifier Evaluation:** V selects the initial classifier C_x corresponding to the hub A. The verifier server V obtains the type $\mathcal{T}_D(i,x)$ and accuracy $a(i,x)$. If the accuracy $a(i,x) \geq \tau$, V transmits $\text{AE}(\mathcal{T}_D(i,x))$ to A. Otherwise, V makes table with type and accuracy $[\mathcal{T}_D(i,x); a(i,x)]$, and sorts it according to accuracy in descending order.

3. **Targeted Classifier Evaluation:** V selects three types with highest accuracy $\mathcal{T}_D(i,x)$, $\mathcal{T}_D(j,x)$, and $\mathcal{T}_D(k,x)$. Further, V evaluates the fingerprint $\mathcal{F}_D$ using

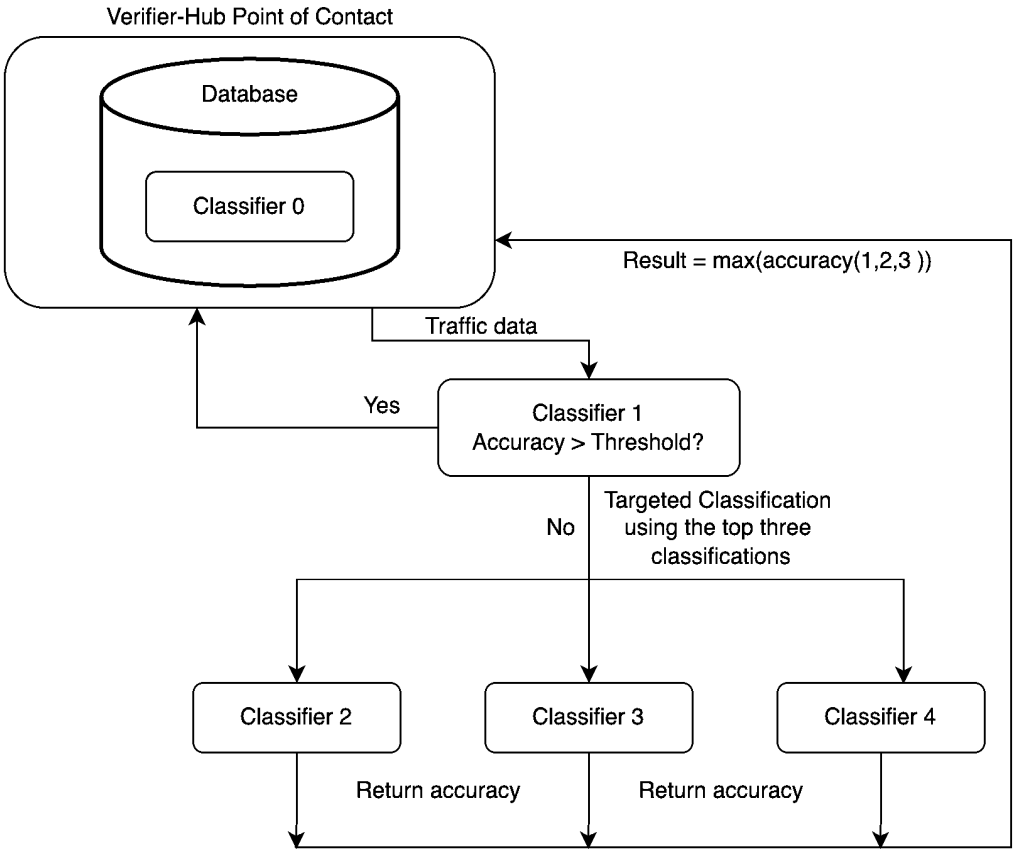

Figure 4.1: Verifier

the three classifiers C_i, C_j, and C_k corresponding to the types with highest accuracy. If any of the accuracy of the classifiers $a(y, y) \geq \tau \ \forall \ y = \{i, j, k\}$, V transmits the corresponding type $\text{AE}(\mathcal{T}_D(y, y))$ to A. Otherwise, repeat the step.

4.2.1 RADTEC: The Protocol

The re- and continuous authentication protocol can be utilized for both kinds of device: the devices re-introduced into the network or the devices constantly present in the network. The intuition is to utilize the cryptographic credentials and observed device type for authentication of the device. A device (D) that is being reintroduced

into the network or a constantly present device (D) that refreshes the credentials, presents the credentials from the previous session to the hub (A). The hub verifies the validity of the credentials and places the device in 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. 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 the captured traffic to the verification server (V). The server V computes the fingerprint of the device type based on the traffic and compares the observed fingerprint of the device type with the fingerprint saved in the database, and finally, returns the classification results to A.. 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. Here 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. the legitimate device with compromised credentials will have to complete the manual initial trust establishment. Formally, the protocol follows the following steps:

1. **Initialization:** The user initiates the initial authentication where the device (D) then transmits the credentials $\mathcal{C}_D := \{\mathcal{K}_D, \mathcal{I}_D\}$ to the hub A, where $\mathcal{K}_D$ is the pre-shared key and $\mathcal{I}_D$ is the identity of D.

2. **Limited Access:** After verification of $\mathcal{C}_D$, A places D on limited access network. In case of failure, the session is terminated by A.

3. **Capture Traffic:** The device (D) establishes the connection to the network

and transmits data to its cloud service. This traffic $\mathcal{T}_D$ is captured by A for a pre-determined time.

4. **Device Type Classification:** AE($\mathcal{T}_D$) is transmitted by A to the verification server (V) on the trusted channel. If the MAC address of the device is unknown, V executes the classification algorithm on $\widehat{\mathcal{T}}_D$ after verifying the integrity and authenticity of the message. After the device has been successfully verified through the classification process, the device type fingerprint $\mathcal{F}_D$ is saved on a database for future authentication.

5. **Device Type Verification:** If the MAC address of the device is known and the device has been previously authenticated or after the device has been successfully classified, V retrieves the stored device type fingerprint $\mathcal{F}_D$ saved on the database corresponding to the identity $\mathcal{I}_D$. Finally, V performs the verification $\widehat{\mathcal{F}}'_D \stackrel{?}{=} \mathcal{F}_D$. and sends the results to A.

6. **Full Access:** If the verification passes D is granted full access to the network according to any deployed policy. Otherwise the session is rejected by A and the credential $\mathcal{C}_D$ is marked compromised.

7. **Continuous-authentication:** Any IoT device in the network with full-access can be re-authenticated at least 'n' times a day by using **Device Type Verification** to maintain the security of the network and to identify a compromised device.

Chapter 5

Security Analysis

In this section, we analyze the RADTEC protocol and the classification techniques used in this paper by discussing the key feature that makes the proposed solution resistant to multiple attacks. Finally, we discuss some additional scenarios where the protocol can be implemented to provide complete security to the system, which involves implementing different authentication techniques based on the behavior of the device.

5.1 Analysis of RADTEC

Initially, a user initiates the authentication of a new device through the use of device credentials that utilize cryptographic techniques to create the initial connection between a device and the hub. RADTEC is capable of identifying whether a device is known or unknown through the use of above mentioned classification techniques. The protocol addresses scenarios where an adversary can: 1. exploit a vulnerable device to inject malicious packets and therefore, poison the network [16], 2. use a vulnerable IoT device to extract sensitive data even if the network packets are encrypted [2], 3. compromise a vulnerable IoT device and actuate the activities that would typically be performed by a different type of device [43]. 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.

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 and re-authenticated in the future. If the device is still compromised during re-authentication, the classifiers will not be able to provide an accurate classification of the device since the device fingerprint will still be different. For example, if an IoT camera is compromised, the classifiers in the verifier will not be able to classify the device as a camera since the fingerprint collected from the device will not be similar to any IoT camera fingerprint used to train the classifiers.

5.2 Analysis of the Classification Technique

We take the approach mentioned in this thesis rather than only identifying a new device based on the MAC address like in [24], where the adversary could easily spoof the MAC address of an already authenticated device and authenticate itself with the hub [36]. In our approach, the device is classified every time it needs re-authentication

irrespective of its previous authentication with the server. The only time we use the MAC address is to check if the device already exists in the database; so, even if the MAC address is spoofed, the classifier in the database will not be able to make a classification with the highest accuracy since the fingerprint of the device does not match with the one already stored in the database, creating a contradiction since the MAC address is the same but the fingerprint of the device is completely different. This would address the case where a vulnerable device is compromised since the traffic pattern and, therefore, the fingerprint of the device will now be different from before. The verifier will automatically be able to send this confirmation to the hub that the MAC addresses match, but the fingerprint does not match, so this device must be an illegitimate device.

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.

5.3 Discussion

We assume that there are three scenarios in which a device needs to be re- and continuous authentication. In this section, we present different scenarios in which a device would need to be authenticated within a network to achieve full access.

When **a new device is first introduced in the network**, the hub collects the device fingerprint and network information, such as the MAC address of the IoT de-

vice, and forwards it to the verifier. The verifier then looks up the MAC address in the database and realizes that the device has not been previously connected. The device type-level classifiers along with the targeted classifiers are used to make an accurate classification of the device, and then, based on the results, the device is granted full access to the network. The device fingerprint is further used to train a classifier and store it in a database for future classifications, as explained previously in our protocol.

However, if **a previously authenticated device moves in and out of the network and requires re-authentication**, the verifier simply collects the fingerprint of the device and its MAC address and uses it to first confirm whether the device with the same MAC address has been previously classified. Then the verifier attempts to make a classification of the device type using the classifier stored in the database, and, based on the classification results, it notifies the hub to give full or limited network access to the device.

Finally, if **a device that constantly remains inside the network requires continuous authentication** and any device can be continuously authenticated up to "n" number of times a day. The process of authentication then would be similar to the re-authentication process.

The protocol addresses all three scenarios by correlating a device MAC address to the ones previously stored in the database, and then providing authentication by the use of the classification techniques applied in this paper. Furthermore, if a vulnerable device is compromised by an adversary and the authentication credentials are stolen, it will generally be the case that the adversary does not know the type of device. However, even if the adversary is able to identify the type of device, the adversary

will not be able to perform any actions using the vulnerable device, since the classification model will notice the change in device fingerprint, resulting in blocking the access given to the device.

It is important to note here that if a device is compromised, it will not be detected until the next time the device needs to be re-authenticated and during that time, the adversary can use the compromised credentials to perform malicious activities. This is acceptable since the security administrator will be able to control the parameters for performing multiple re-authentications for a device in a certain amount of time. This process is similar to the frequency at which the key revocation process operates, which depends on the security requirements of different devices and networks [9].

Chapter 6

Implementation

In this section, we discuss the selection of the data set and the process for device identification. Additionally, we present the implementation techniques used to classify IoT devices such as data preprocessing involving, data cleaning and splitting, standardizing features, numerical imputation, and feature engineering. Finally, we discuss the training of the classifiers used to classify the type of IoT devices.

6.1 Dataset

Traffic between all devices listed in the table and the access point was acquired from data collected at the University of New South Wales [37]. The data collected include more than 28 IoT devices such as cameras, motion sensors, health monitors, appliances, etc. A subset of data collected over the period of six months has been made available as open-source. We chose 15 devices as shown in Table 6.1 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 ($n \times f$). We further classified the different devices according to the type of device they are and assigned them a class for the model training and testing process.

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 Electric and Lighting	3
Light Bulbs LiFX Smart Bulb	Smart Electrical and Lighting	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

Table 6.1: List of IoT devices and their classes

6.2 Device Identification

The device identification process is done by utilizing the device fingerprint collected by the hub. The list of devices mentioned above consists of certain types of devices, but not all types of IoT devices. However, our process for developing the ML model is scalable. New categories of IoT devices can simply be trained on the combined classifier and targeted classifiers through the use of device packet capture.

6.2.1 Algorithm selection

Based on previous work and approaches [4, 41, 7, 24], we decided to use the following five classifiers: random forest, k-nearest neighbors, gradient boosting, naïve Bayes, and support vector machine. However, we improved the previous work that classifies the IoT device and performs only the IoT or non-IoT classification through higher accuracy and a more efficient classification process. In our research, we not only classified whether the device was an IoT device or not, but also performed a further classification of the type of IoT device such as smart cameras, home assistants, smart switches and plugs, etc.

6.2.2 Data Pre-processing

For better classification results the data was preprocessed by using several techniques. Following are the techniques used in this paper,

6.2.2.1 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 packets with source ethernet addresses that were not required in our model. 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 such packets present in the data. The empty columns representing empty values for features were also removed since they do not provide any contribution to the classification process. Finally, the entire dataset was labeled in different classes for training and testing.

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 X (features) and Y (labels). Then both X and Y were randomly split into train and test data sets in a ratio of 80% and 20% respectively.

6.2.2.2 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,

$$z = \frac{v - m}{s} \tag{6.1}$$

where, v = values of the sample dataset, m = mean of training samples and s = standard deviation of the training samples

6.2.2.3 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 affect the amount of data needed to make accurate classifications. Therefore, 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.

6.2.2.4 Feature Engineering

First, we use a random forest search classifier to extract feature importance scores from the 19 features shown in Fig.6.1. After calculating the importance scores for

the features, we set a threshold of 0.05 for the importance score of the features and eliminated all features with importance scores below the threshold.

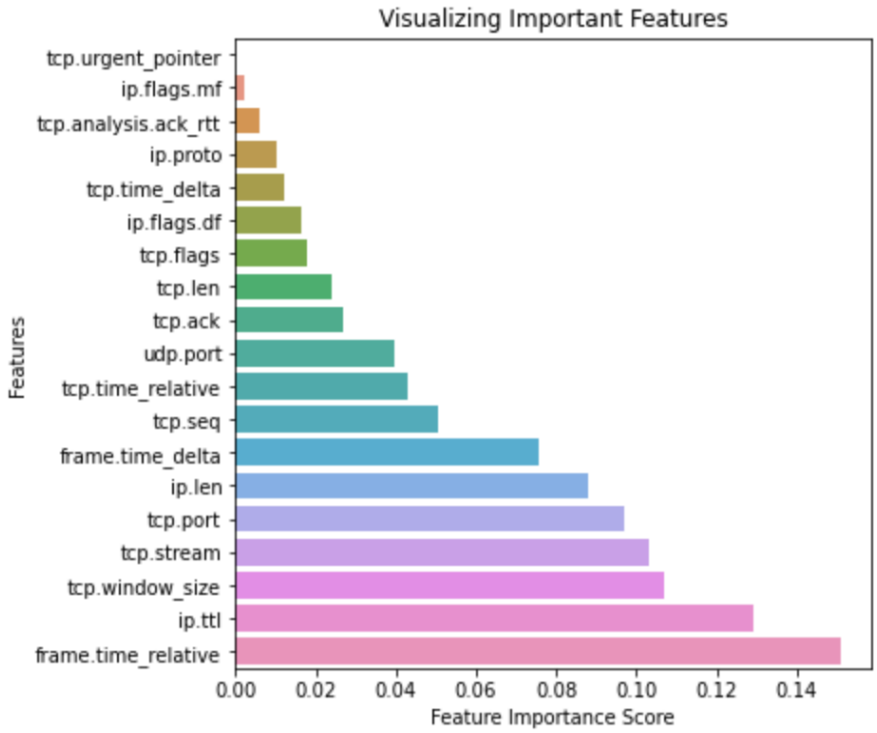

Figure 6.1: Feature Importance Scores

The data that we used to classify the type of IoT devices had several features that were not useful for making predictions. Such features included tcp.urgent_pointer, ip.flags.mf, tcp.analysis.ack_rtt, ip.proto, tcp.time_delta, ip.flags.df, tcp.flags, tcp.len, tcp.ack, udp.port, tcp.time_relative, tcp.seq. These features were removed from the dataset because they did not provide any valuable contribution, negatively affected the required runtime and memory usage, and reduced the accuracy of the classifiers used.

6.3 Training and Testing

After collecting and pre-processing the dataset, 80% of it was used to train each of these five classifiers individually: Random Forest Classifier (RFC)[32], K-Nearest Neighbors Classifier (KNN)[30], Support Vector Machine Classifier (SVM)[33], Gradient Boosting Classifier (GBC)[29], and Gaussian Naive Bayes Classifier (GNB)[31]. The training process was performed using the *fit* method provided by *sklearn*, which fits the model onto the data to later provide predictions [28]. During the training process, the time taken to fully train each model was recorded and is shown in Table 6.2.

During 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 [34]. Finally, the time required to train each model was calculated and the average time required to classify a single data point is shown in Table 6.2

Model	**Train Time**	**Test Time**
Gaussian Naive Bayes Classifier	50 ms	0.23 ms
Random Forest Classifier	18.17 sec	0.63 ms
K-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

Table 6.2: Time required for training each model and the average time required to classify one device.

Chapter 7

Results

In this chapter, we present the results of the classification algorithms employed in this thesis to identify the device type-level classification. To achieve the most accurate classification of all the IoT devices, we trained five different models including Random Forest Classifier (RFC), K-Nearest Neighbors (KNN), Support Vector Machine Classifier (SVM), Gradient Boost Classifier (GBC) and, Gaussian Naive Bayes (GNB) classifier and then evaluated their performance. The metrics used to analyze each model performance were the F1 score and the accuracy score.

7.1 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 true correct positives and the total number of instances predicted to be positive and is given by Eq. 7.1.

$$Precision = \frac{Number\ of\ True\ Positives}{Number\ of\ True\ Positives\ +\ Number\ of\ False\ Positives} \tag{7.1}$$

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 Eq. 7.2.

$$Recall = \frac{Number\ of\ True\ Positives}{Number\ of\ True\ Positives\ +\ Number\ of\ False\ Negatives} \tag{7.2}$$

The value of the F1 score can range between 0 and 1 where 1 is the highest score a model can achieve and the values of both precision and recall are the highest. The formula for the F1 score is given by Eq. 7.3.

$$F1\ Score = 2 \times \frac{Precicion \times Recall}{Precision + Recall} \tag{7.3}$$

We calculate the F1 score using the evaluation metric library provided by *sklearn .metrics* [1]. The F1 score for each class within each model is plotted in the Fig. 7.1 where class 0 represent non-IoT devices, class 1 represents Home Assistants, class 2 represents Smart Camera, class 3 represents Smart Bulb & Smart Electrical, class 4 represents Smart Sensors, and class 5 represents Smart Speaker. The highest average F1 score for all classes was provided by the Random Forest Classifier (RFC) where it can further seen that the model is near perfect for differentiating between an IoT and a non-IoT device.

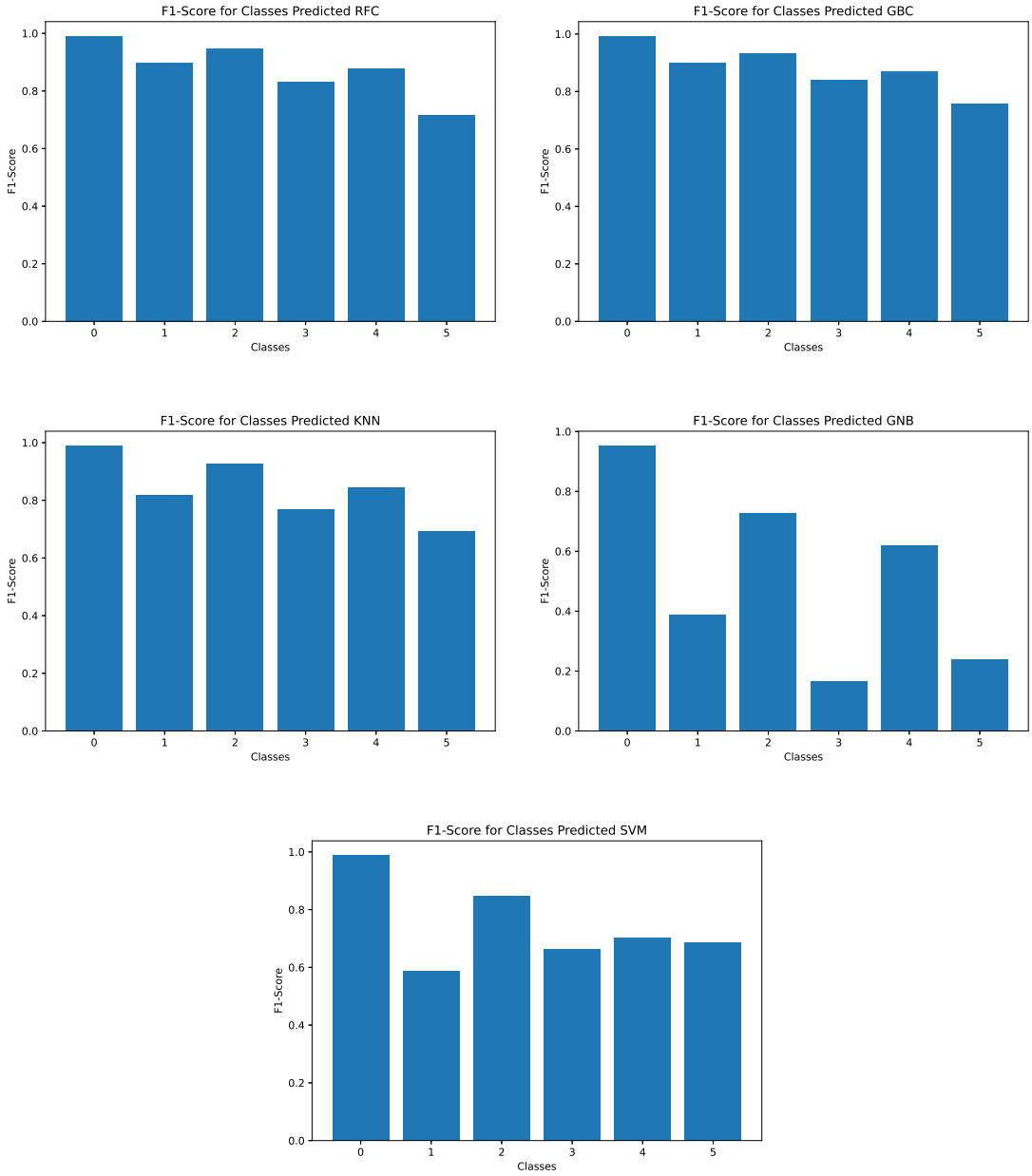

Figure 7.1: F1 scores for Random Forest Classifier (RFC), Gradient Boost Classifier (GBC), K-Nearest Neighbors Classifier (KNN), Support Vector Machine Classifier (SVM), and Gaussian Naive Bayes Classifier (GNB).

7.2 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 predic-

tions made by the classifier and the total number of predictions to be made (Eq. 7.4). Additionally, the percentage of this score can be calculated to obtain the accuracy of a classifier in terms of a percentage.

$$Accuracy\ Score = \frac{Number\ of\ correct\ predictions}{Number\ of\ total\ predictions} \tag{7.4}$$

The function to calculate the accuracy of our machine learning models used to perform multi-class classification was provided by the *sklearn.metrics* library [34]. After making the predictions, we established that the Random Forest Classifier (RFC) was the most accurate model for making predictions with an accuracy of 95.2%. The second most accurate classifier was the Gradient Boost Classifier (GBC) with an accuracy of 94.8%, then the K-Nearest Neighbors Classifier (KNN) with accuracy 93.3%, Support Vector Machine Classifier (SVM) with accuracy of 88.3%, and finally the Gaussian Naive Bayes Classifier (GNB) with accuracy of 76.8%.

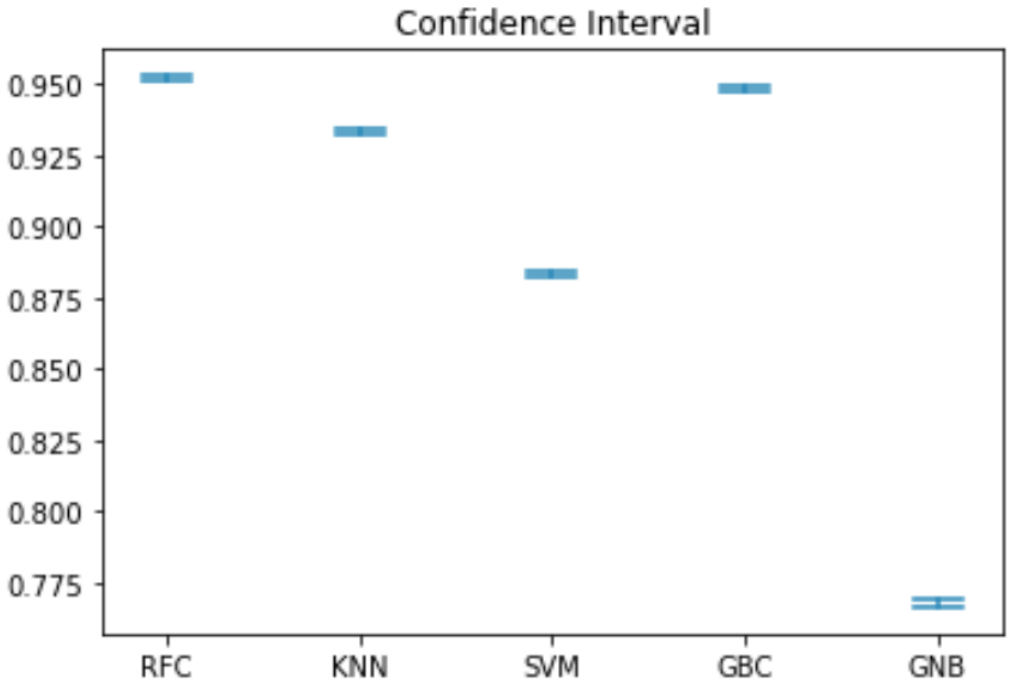

Figure 7.2: Accuracy vs Model plot for Confidence Interval

Furthermore, using the accuracy score, we also determine the 95% confidence interval for each classifier. The method essentially provides an upper bound and a lower bound for accuracy, which represents all the possible values accuracy can have. Therefore, when any device needs classification, there will be a 95% likelihood for it to be classified will lie between that range and is given by the following equation,

$$95\%\ Confidence\ Interval = 1.96\ \times \sqrt{\frac{(accuracy\ \times\ (1\ -\ accuracy))}{n}} \tag{7.5}$$

The upper and lower bounds of the 95% confidence interval for each classifier are shown in the Table7.1 and its plot is shown in Fig.7.2.

Model	**CI Upper Bound**	**CI Lower Bound**
Random Forest Classifier	0.9527	0.9513
K-Nearest Neighbors Classifier	0.9338	0.9322
Support Vector Machine Classifier	0.8840	0.8820
Gradient Boosting Classifier	0.9487	0.9473
Gaussian Naive Bayes Classifier	0.7694	0.7666

Table 7.1: Upper bound and lower bound of 95% Confidence Interval (CI)

Chapter 8

Conclusion and Future Work

In this thesis, we proposed the RADTEC protocol as a solution to improve the security of IoT devices. Through the use of several machine learning and cryptographic techniques, RADTEC addresses the events in which an adversary can exploit a vulnerable device to: 1. capture network traffic, 2. poison the network by injecting malicious packets, and 3. actuate a device to perform activities of some other type of device. We then performed the security analysis of our protocol and presented the implementation of device type level classification that utilizes cross-layer data, including network, data link, transport, and application, and it was carried out using five classifiers, where the performance of each classifier was measured using the accuracy and F1 score evaluation metrics.

We used the Random Forest Classifier, Gradient Boost Classifier, K-Nearest-Neighbors Classifier, Support Vector Machine Classifier, and the Naive Bayes Classifier to identify whether a device in the network is an IoT device or not, and then further classified the type of IoT device. Of all models, the Random Forest Classifier was able to make the most accurate prediction with an accuracy score of 95.2% and the highest average F1 score. Finally, the time required to train the models and the average time taken to classify a single data point were recorded for all classifiers.

Gaussian Naive Bayes Classifier was found to be the fastest classifier for training and testing; however, we prefer to use the Random Forest Classifier, as it still required less time than most other models for training and testing and provided the most accurate results.

8.1 Future Work

While we have provided a solution to improve IoT security, there are other important issues that still need to be addressed. Therefore, in our future work we plan to train classification models for several more types of IoT devices. We will also collect more data and include additional features from the packet capture for each type of device to provide more accurate classifications. In addition to the machine learning classifiers used in this thesis, we will perform device type level classification using several other types of classifiers and apply model fine-tuning techniques to improve their performance.

Bibliography

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

[2] 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*, pages 207–218, 2020.

[3] T. Ahmad and D. Zhang. Using the internet of things in smart energy systems and networks. *Sustainable Cities and Society*, page 102783, 2021.

[4] 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*, 22(3):1646–1685, 2020.

[5] 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].

[6] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *Proc. of CRYPTO*, pages 531–545. Springer, 2000.

[7] 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*, pages 1–9. IEEE, 2020.

[8] R. Chatterjee, P. Doerfler, H. Orgad, S. Havron, J. Palmer, D. Freed, K. Levy, N. Dell, D. McCoy, and T. Ristenpart. The spyware used in intimate partner violence. In *Proc. of IEEE Symposium on Security and Privacy (SP)*, pages 441–458, 2018.

[9] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, W. T. Polk, et al. Internet x. 509 public key infrastructure certificate and certificate revocation list (CRL) profile. *RFC*, 5280:1–151, 2008.

[10] CVE. IoT search results. `https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword={IoT}`. [Online; accessed 20-Mar-2022].

[11] 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*, page 667. ACM, 2018.

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

[13] N. Ghose, L. Lazos, and M. Li. In-band secret-free pairing for cots wireless devices. *IEEE Transactions on Mobile Computing*, 2020.

[14] N. Z. Gong, A. Ozen, Y. Wu, X. Cao, R. Shin, D. Song, H. Jin, and X. Bao. Piano: Proximity-based user authentication on voice-powered internet-of-things

devices. In *Proc. of International Conference on Distributed Computing Systems ICDCS*, pages 2212–2219. IEEE, 2017.

[15] 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*, pages 105–122, 2019.

[16] S.-Y. Hwang and J.-N. Kim. A malware distribution simulator for the verification of network threat prevention tools. *Sensors*, 21(21):6983, 2021.

[17] IDC. IoT growth demands rethink of long-term storage strategies, says idc, Jul 2020.

[18] J. Kesavadev, B. Saboo, M. B. Krishna, and G. Krishnan. Evolution of insulin delivery devices: from syringes, pens, and pumps to DIY artificial pancreas. *Diabetes Therapy*, 11:1251–1269, 2020.

[19] R. S. Lee. Smart city. In *Artificial Intelligence in Daily Life*, pages 321–345. Springer, 2020.

[20] J. Liu and W. Sun. Smart attacks against intelligent wearables in people-centric internet of things. *IEEE Communications Magazine*, 54(12):44–49, 2016.

[21] 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*, 9(1):298–320, 2021.

[22] Y. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N. O. Tippenhauer, J. D. Guarnizo, and Y. Elovici. Detection of unauthorized IoT devices using machine learning techniques. *arXiv preprint arXiv:1709.04647*, 2017.

[23] 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].

[24] 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)*, pages 2177–2184. IEEE, 2017.

[25] 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)*, pages 756–767. IEEE, 2019.

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

[27] P. Robyns, E. Marin, W. Lamotte, P. Quax, D. Singelée, and B. Preneel. Physical-layer fingerprinting of LoRa devices using supervised and zero-shot learning. In *Proc. of ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 58–63, 2017.

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

[29] Scikit-Learn. Sklearn.ensemble.gradientboostingclassifier. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html. [Online; accessed 22-Mar-2022].

[30] Scikit-Learn. Sklearn.ensemble.kneighborsclassifier. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html. [Online; accessed 22-Mar-2022].

[31] Scikit-Learn. Sklearn.ensemble.naivebayes.gaussiannb. https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html. [Online; accessed 22-Mar-2022].

[32] Scikit-Learn. Sklearn.ensemble.randomforestclassifier. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. [Online; accessed 22-Mar-2022].

[33] Scikit-Learn. Sklearn.ensemble.svm.svc. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html. [Online; accessed 22-Mar-2022].

[34] Scikit-Learn. sklearn.metrics.accuracy_score. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html. [Online; accessed 24-Mar-2022].

[35] H. Shafagh and A. Hithnawi. Poster: come closer: proximity-based authentication for the internet of things. In *Proc. of Annual International Conference on Mobile Computing and Networking*, pages 421–424, 2014.

[36] Y. Sheng, K. Tan, G. Chen, D. Kotz, and A. Campbell. Detecting 802.11 mac layer spoofing using received signal strength. In *Proc. of IEEE INFOCOM*, pages 1768–1776. IEEE, 2008.

[37] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman. Classifying IoT devices in smart environments us-

ing network traffic characteristics. *IEEE Transactions on Mobile Computing*, 18(8):1745–1759, 2018.

[38] 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.

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

[40] 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*, 152(2):138–143, 2005.

[41] 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*, 35(5):41–49, 2018.

[42] Z. Yi, F. Xie, Y. Tian, N. Li, X. Dong, Y. Ma, Y. Huang, Y. Hu, X. Xu, D. Qu, et al. A battery-and leadless heart-worn pacemaker strategy. *Advanced Functional Materials*, 30(25):2000477, 2020.

[43] J. Yun, I.-Y. Ahn, J. Song, and J. Kim. Implementation of sensing and actuation capabilities for IoT devices using oneM2M platforms. *Sensors*, 19(20):4567, 2019.

[44] J. Zhang, Z. Wang, Z. Yang, and Q. Zhang. Proximity based IoT device authentication. In *Proc. of IEEE INFOCOM*, pages 1–9. IEEE, 2017.