ANALYSIS OF PRIVACY THREATS IN INTERNET OF MEDICAL THINGS (IoMT) USING MACHINE LEARNING

by

Tilak Bahadur Basnet

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science

Computer Science

At

The University of Wisconsin-Whitewater

December, 2022

Graduate Studies

The members of the Committee approve the thesis of

Tilak Bahadur Basnet presented on December 2, 2022

Dr. Jiazhen Zhou, Chair

Dr. Zachary Oster

Dr. Chandra Sharma

Analysis of privacy threats in Internet Of Medical Things (IoMT) using machine learning

By

Tilak Bahadur Basnet

The University of Wisconsin-Whitewater, 2022 Under the Supervision of

Dr. Jiazhen Zhou

ABSTRACT

The growing market for Internet of Medical Things (IoMT) promises new conveniences for consumers while presenting new challenges for preserving consumers' privacy. Most of the IoMT devices have sensors that keep capturing users' activities and transmit information about these activities on the internet. In this thesis, we demonstrate that machine learning schemes could be employed by an Internet Service Provider (ISP) or an intentional attack to infer device name/ type and privacy sensitive user activities by analyzing internet traffic generated from commercially-available IoMT devices, even though those devices use end-to-end transport-layer encryption. To better protect the privacy of IoMT device users, we proposed a traffic shaping scheme. Our experiments showed that traffic shaping significantly reduces an adversary's ability to accurately identify devices owned by users or detect genuine user activities/interactions. Hence, it can effectively and practically mitigate many privacy risks associated with IoMT devices.

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere gratitude and appreciation to my academic advisor Dr. Jiazhen Zhou for his advice, support, motivation, and patience throughout the last one and half 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 would like to thank Dr. Zachary Oster and Dr. Chandra Sharma for serving on my master's committee and for providing insightful and valuable input that helped me make this thesis better.

Also, I would like to thank all of the faculty members in the Computer Science Department for their great comprehension, and collaboration.

I am extremely grateful to my mother Ganga Maya Basnet, and my father Rajendra Kumar Basnet for their love, prayers, support, and guidance throughout my life; you have always been my source of inspiration, confidence, and success. Your love, care and support give me the strength to overcome any difficulties that I may face.

I dedicate this thesis to my family.

Thanks for giving me the opportunity to study in the U.S.

TABLE OF CONTENTS

Acknowledgments iii

List of Tables iv

List of Figures v

Chapter 1: Introduction 1

Chapter 2: Background and Related Work 3

Chapter 3: Models and Preliminaries 8

3.1 Privacy Attack Model 8

3.2 Machine Learning Models Used for Privacy Attacks 9

3.2.1 Supervised Learning 9

3.2.1.1 K-Nearest Neighbors (KNN) [6] 10

3.2.1.2 Random Forest Classifier (RFC) [9] 11

3.2.1.3 Support Vector Machine Classifier (SVM) [10] 12

3.2.1.4 Gradient Boost Classifier (GBC) [5] 13

3.2.1.5 Gaussian Naive Bayes Classifier (GNB) [4] 13

3.3 Model performance metrics 14

3.3.1 F1 Score [2] 14

3.3.2 Accuracy Score [1] . . . 15
3.4 Experimentation Setup and Data Collection . . . 16
3.4.1 Data collection environment . . . 16
3.4.2 Separate traffic into per-device packet streams from overall network traffic . 17
3.4.2.1 Separate entire traffic into packet streams . . . 17
3.4.2.2 Label packet streams by IoMT device type . . . 18

Chapter 4: Device and Activity Identification Using Machine Learning Classifiers . . . 19

4.1 Training dataset preparation from the PCAP files . . . 19
4.2 IoMT device identification/fingerprinting . . . 22
4.2.1 Using DNS Queries . . . 22
4.2.1.1 FitBit Smartwatch . . . 23
4.2.1.2 Kardia EKG Monitor . . . 23
4.2.1.3 QardioArm Smart Blood Pressure Monitor . . . 24
4.2.2 Using Machine Learning on traffic metadata . . . 24
4.2.2.1 Feature Selection . . . 25
4.2.2.2 Data Pre-processing . . . 25
4.2.2.3 Training and Testing . . . 27
4.2.2.4 Classifier Performance . . . 28
4.3 IoMT event/activity identification . . . 30
4.3.1 Examine traffic rates . . . 30
4.3.2 Using Machine Learning on traffic metadata . . . 32
4.3.2.1 Feature Selection . . . 32

4.3.2.2 Data Pre-processing 33
4.3.2.3 Training and Testing 33
4.3.2.4 Classifier Performance 34

Chapter 5: Defense Against Machine Learning Based Privacy Attacks 36
5.1 Traffic Shaping 36
5.2 Impact of Traffic Shaping on IoMT Device Identification 38
5.3 Impact of Traffic Shaping on Activity/Event Classification 40
5.4 Summary of Traffic Shaping Implementation 42

Chapter 6: Conclusion and Future Work 43
6.1 Conclusion 43
6.2 Future Work 44

LIST OF TABLES

4.1 Application layer and corresponding protocol, feature used 20

4.2 IoMT device and corresponding DNS Query 22

4.3 IoMT device and corresponding class 26

4.4 Fitbit activities and their corresponding classes 33

LIST OF FIGURES

2.1 Partial Traffic Reshaping implemented in [39] 7

3.1 Supervised Learning 10

3.2 K-Nearest Neighbors Classifier 11

3.3 Random Forest Classifier 12

3.4 Support Vector Machine Classifier 12

3.5 Gradient Boost Classifier 13

3.6 Simple Gaussian Naive Bayes Classifier 14

3.7 Data collection environment 16

4.1 Feature importance scores using Random Forest Classifier 21

4.2 FitBit DNS Query 23

4.3 Kardia DNS Query 23

4.4 QardioArm DNS Query 24

4.5 VisualBeat DNS Query 25

4.6 F1 scores of device identification for GBC, GNB, KNN, RFC and SVM 28

4.7 Accuracy of device identification for different ML models 29

4.8 Kardia EKG Monitor Traffic when device events changed 31

4.9 Fitbit Smartwatch Traffic when device events changed 32

4.10 F1 scores of Fitbit activity/event classification for GBC, GNB, KNN, RFC and SVM 34

4.11 Accuracy of Fitbit activity/event classification for different ML models 35

5.1 Inbound traffic traces of FitBit Smartwatch before and after traffic shaping 37

5.2 Outbound traffic traces of FitBit Smartwatch before and after traffic shaping 38

5.3 Accuracy of device fingerprinting after traffic shaping 39

5.4 Average training time of device identification classifiers 40

5.5 Accuracy of Fitbit activity/event identification after traffic shaping 41

5.6 Average training time of Fitbit event/activity classifiers 42

CHAPTER 1
INTRODUCTION

The Internet of Things ("IoT") refers to devices that are capable of interacting with human or surrounding environments, and can connect to the Internet for sending and receiving data. This definition includes a variety of Internet-connected home-based, medical devices increasingly deployed in homes, hospitals and health care centers. The number of Internet of Things (IoT) devices is projected to grow from 13.8 billion in 2021 to 30.9 billion units in 2025. Internet of Medical Things (IoMT) are designed to record user data which are privacy sensitive and tend to collect vast amounts of personal information. They provide greater efficiency to healthcare as they enable end users to share information remotely and receive timely feedback from health care providers. Specifically, the rapid development of IoMT has improved the traditional medical systems, such as disease diagnosis and analysis. The health data collected in IoMT can be used by researchers to diagnose and predict diseases. Such information is private and could indicate patients' health status, medical data, preferences, behaviors and habits. In recent years, older adults have become wide adopters of IoMT devices even with little or no understanding of how the technology works or how the information is shared which is posing serious privacy risks for older adults. Such sensitive health information might be misused by device manufacturers, intercepted by adversaries and network observers if being transmitted in plain/clear text which is not very common anymore. However, even when IoMT devices encrypt data transmitted to the cloud, network observers could inspect traffic metadata like packet size, packet inter arrival time, protocols used in different network layers, timestamp, direction of traffic flow (incoming/outgoing) etc. to obtain and misuse users' private information such as users' behavior and interaction, pattern of device usage based on its activity. Users trust IoMT device manufacturers to protect their privacy and ensure security. But in most cases they do not have the technical ability to verify that these protections are in place.

Most of the previous work includes the studying of the smart home privacy and security which

deals with smart home devices like Amazon Echo, Nest Security Camera, Belkin WeMo Switch etc. However, given the gradually wider adoption of IoMT devices by any age group of users and limited research in understanding the impacts of these devices on their privacy, research in IoMT device privacy protection is becoming more critical. The main goals of this research are two-fold. One is to evaluate the effectiveness of machine learning classifiers in privacy attacks that identify the IoMT device type and activities. The other goal is to research the defense on the privacy attacks against IoMT devices via traffic shaping techniques.

The major contributions of our work include:

- We present machine learning based device fingerprinting and event/activity identification of IoMT devices. The proposed technique utilizes cross-layer data including network, data link, transport, and application layer data. To make the classification algorithms robust, we identified top 5 features to train the classifier.

- We perform extensive experimentation for our machine learning based device type and event/activity identification technique to show the performance of various machine learning classifiers including K-Nearest Neighbors (KNN) [6], Random Forest (RF) [9], Support Vector Machine (SVM) [10], Gradient Boosting Classifier (GBC) [5] and Gaussian Naive Bayes (GNB) [4]. Our experimentation shows that the K-Nearest Neighbor (KNN) outperforms the rest of the four classifiers.

- Device events or user activities having similar traffic rates/patterns could not be differentiated by traffic rate observation techniques. We develop a classifier that is capable of distinguishing events/activities those events/activities.

- We implement a traffic shaping scheme to obscure the privacy information that was exposed with traffic rate traces and its metadata. We conducted simulations on the traffic shaping scheme and showed its effectiveness that is demonstrated through a largely reduced identification accuracy in comparing cases without defense.

CHAPTER 2
BACKGROUND AND RELATED WORK

Privacy attacks on IoT devices using both machine learning based classification and other non-machine-learning techniques have been studied extensively. These research have mainly focused on differentiating several types of IoT devices from non-IoT devices, fingerprinting IoT devices, and tracing user activities from network traffic intercepted. More recently, there has been some work about defending these machine learning based privacy attacks, mainly using traffic shaping algorithms to prevent the machine learning algorithms from identifying the IoT devices and user activities correctly. In this chapter, we present recent research in this area and their differences from this thesis research work.

There has been research about attacks against privacy in IoT devices using non-machine-learning techniques. Researchers in [16] collected data from generic smart home devices that include a Sense sleep monitor, a Nest Cam Indoor security camera, a WeMo switch and an Amazon Echo. To identify the devices that a consumer owns, they mapped the Domain Name System (DNS) queries associated with each stream to a particular device. For example, the Nest Cam queried domains from dropcam.com, while the Sense sleep monitor queried domains from hello.is (manufacturer of Sense sleep monitor). They were able to identify the consumer devices from the DNS queries, which can be a serious privacy violation. However, this approach has its limits. First, some domain names might not contain any hint about the server or manufacturer. Second, it might not always be the case that we can map DNS query to a particular device because multiple devices from the same manufacturer might communicate with the same server, making device identification using DNS more difficult.

In the same research, the authors also successfully demonstrated that traffic rates can reveal sensitive information about a user's online activities by simply plotting send/receive traffic rates of the streams. For instance, a traffic spike from the sense sleep monitor in the late evening likely

corresponds to when the user went to sleep. Even though the traffic payload was completely encrypted, they found that variation on network traffic rates can reveal potentially sensitive user interactions. They concluded that encryption alone does not provide adequate privacy protection for smart homes.

It is worthy to note that, the smart home IoT devices examined in [16] were of the limited-purpose nature: they only have two distinct (binary) events like went to bed or NOT on Sense Sleep Monitor, switch turned ON/OFF, question asked to Amazon Echo or NOT, motion or idle on Nest Camera. For each device, one event was supposed to cause traffic to spike and another to lower. However, this simple pattern is often not the case for the IoMT devices that we studied. In fact, it might be difficult to distinguish two events if they both have similar traffic patterns (like both cause traffic to spike or low). For instance, activities like running and workout both cause FitBit Smartwatch traffic to spike. As a result, we could not guarantee that simply plotting network traffic rates can reveal the exact user activities performed on IoMT devices.

Due to limitations of non-machine-learning techniques cited above, different machine learning techniques have been presented as a strategy for device identification and activity classification. Researchers in [27] used machine learning based techniques utilizing source, destination IP addresses and port numbers as feature sets to classify the devices into two categories (IoT devices and non-IoT devices) and their corresponding classes. They proposed a multi-stage classifier: in the first stage, the classifier can distinguish between traffic generated by IoT devices like Withings Home Security Camera, Wemo Motion sensor etc. and non IoT devices like Dell PC, Galaxy S4 smartphone etc. In the second stage, each IoT device is associated with a specific device class. Their model was able to associate LG G2 and Galaxy S4 with Smartphone, Dell Optiplex 9020 with PC, Lenovo X260 with Laptop, LG Urban with Smartwatch, Wemo F7C028uk with Motion Sensor and so on. The overall IoT classification accuracy of their model was 99.281%. Similarly, another research group in [19] proposed two classifiers that were capable of identifying a device as IoT or non-IoT, in a short time scale, and with accuracy of 95%. The first classifier was a logistic regression classifier based on traffic features and the other one was based on features retrieved

from DHCP packets.

Although both research groups provided a highly accurate classification model, the scope of their research was to classify devices as IoT or non-IoT devices and did not work on fingerprinting the exact devices themselves. Since the passive network observer could use traffic metadata attack to infer the actual devices being used by the users, which is also a potential privacy vulnerability for the users. For example, consumers might not want an Internet Service Provider (ISP) or passive network observer knowing they own an IoT blood sugar monitor or pacemaker. In fact, simply knowing a device a user owns could have unwanted advertising implications, or it enables other people to know one's potential health status and problems, medical history etc.

Researchers in [32] conducted experiments to identify the IoT-device interactions that can be inferred from metadata of encrypted traffic. They trained a random forest machine learning classifier with a set of features which are timing statistics of the traffic with respect to packet sizes and inter-arrival times. The statistical properties they considered were min, max, mean, deciles of the distribution, skewness, and kurtosis. They have cited that they did not attempt to produce the most performant classifiers according to metrics such as F1 score, rather, they used F1 score metrics to understand whether device activities are inferable. They considered an activity as inferable when its F1 score is greater than 0.75, which is not clear on its own. In our work, we attempt to train multiple classifiers using statistical measures min, max, sum ,mean and standard deviation of frame length and propose one with highest accuracy as the most appropriate model for event/activity identification.

The growing market for smart IoMT devices promises new conveniences for consumers while presenting new challenges for preserving consumers' privacy. Most of the smart IoMT devices have sensors that keep capturing users' activities and transmit information about these activities on the internet. As discussed in previous research works cited above, we saw that an ISP or other network observer can infer device name and type, and privacy sensitive user activities by analyzing internet traffic generated from IoT devices even when the devices use end-to-end transport-layer encryption. The effectiveness of this attack across IoT devices motivates the development of tech-

niques for protecting user privacy. In the following section, we discuss existing defense techniques on machine learning based attacks and present how our work differs from theirs.

Prior work [17] proposed traffic shaping using independent link padding (ILP). Independent Link Padding involves shaping traffic rates to match a predetermined rate or schedule, thereby exposing no information about device and its behavior to an adversary. That is the concept of sending fixed-size packets at a constant rate independent of the underlying device traffic. They used OpenVPN on an Amazon EC2 instance and is a necessary component of their implementation. Similarly, [15] also proposed the Stochastic Traffic Padding in which both upload and download traffic are shaped equivalently, and additional periods of equivalent shaping are injected randomly so that an adversary could not differentiate the user activities and injected traffic from real user activities respectively. The former suffers long network latency, however, the latter impose no additional network latency and relatively little bandwidth overhead, but it would be even more difficult for an adversary to distinguish the false and real user activities (and eventually device event/user activities) if we could inject false traffic at regular time intervals. In our work, we simulate the traffic shaping technique without using VPN and injecting a minimum volume of false traffic at regular time intervals to significantly reduce the confidence of network observers to accurately identify IoMT device and user activities.

Prior work [39] proposed a new low-cost defense system-PrivacyGuard to address the practical limitations like network bandwidth, maximum package injection etc. of existing defense techniques. The PrivacyGuard enables users to significantly reduce the private information leaked through IoT device network traffic data. Their model employs intelligent deep convolutional generative adversarial networks (DCGANs)-based IoT device traffic signature learning, long short-term memory (LSTM)-based artificial traffic signature injection, and partial traffic shaping to obfuscate private information that can be observed in IoT device traffic traces. The main disadvantage of their partial traffic shaping implementation is that it still exposes changes in traffic rate spikes as shown in Figure 2.1 below.

Any passive network observer could easily see the traffic rate pattern by simply plotting the IoT

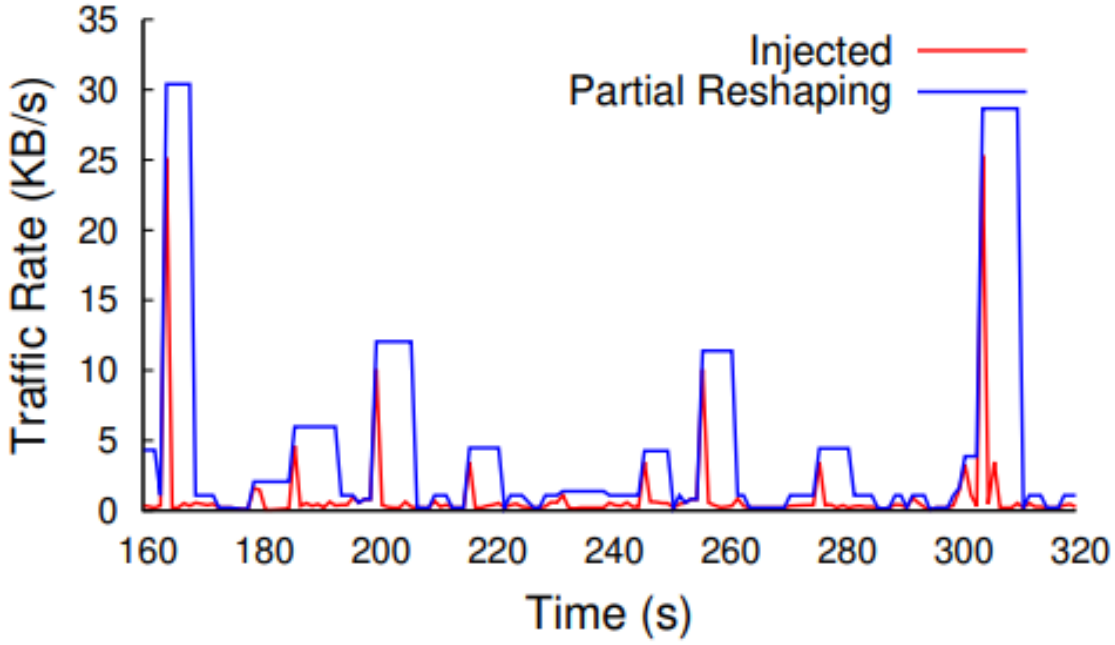

Figure 2.1: Partial Traffic Reshaping implemented in [39]

device traffic. However, in our work we simulate the traffic shaping implementation that makes entire traffic to be of the same sized such that it hides the changes in traffic rate spikes.

While all the above tasks make important contributions, they did not undertake fine-grained characterization and classification of IoT devices. Furthermore, they did not develop models that enable IoT device and event classification based on their statistical characteristics of network traffic. Therefore, there has been extreme significance in the study and research of various privacy attacks on IoMT devices and user activities. In this research, we perform attacks to identify the users' devices and activities and propose the traffic shaping based solution to evade machine learning classifiers from identifying devices and their corresponding events.

CHAPTER 3
MODELS AND PRELIMINARIES

In this chapter, we first present the basic privacy attack ideas. Then we introduce the common machine learning models and evaluation metrics that are used in the attacks.

3.1 Privacy Attack Model

Our research focuses on the abilities of a passive network observer with access to traffic to and from the users' home having IoMT devices. The adversary goals are twofold: (1) to identify the IoMT devices that users are using; (2) to infer user activities using those devices based on the device network traffic observed.

The traffic packet contents are normally encrypted. As a result, the adversary must rely on traffic rate and packet header metadata like packet length, packet inter arrival time, ports accessed by traffic connection, protocols used in different network layers, timestamp, direction of traffic flow (incoming/outgoing) etc. to identify IoMT devices and user activities. Once an adversary identifies a device and knows its purpose, functionalities, device states then the adversary could utilize the change in traffic rates or its variation to infer the user interactions on device at a particular time. Generally speaking, encrypted communications have gradually become the standard way of securing IoMT. Even though, by completely ignoring the packet contents, our traffic metadata privacy attack indicates that sensitive information about users' devices and activities are still at high risk.

At a high level, the adversary leverages devices' known specific functionality to map changes in traffic rates to user activities. For instance, an adversary might first identify that a particular traffic flow is from a blood pressure monitor. Blood pressure monitors generally have only two states(either idle or measuring), so if the flow indicates a spike in traffic rate at a particular time,

the adversary can infer that the user was measuring the blood pressure. Further research could allow passive network observers to infer higher order behavior, such as whether the user has a high or low blood pressure using machine learning techniques. For example, if a particular user measures blood pressure multiple times every day, the user might have blood pressure disease which can be used by an observer or third party for profiling or advertisement purposes.

The device identification and activity identification based on traffic data often involves use of machine learning algorithms that are introduced in the next section.

3.2 Machine Learning Models Used for Privacy Attacks

In this section, we describe supervised learning and five different types of supervised machine learning algorithms used to classify the IoMT device and their corresponding events/activities in the network. Based on previous work and approaches [19], [26], [33], [34], we decided to use five machine learning classifiers, namely K-Nearest Neighbors (KNN), Random Forest Classifier (RFC), Support Vector Machine Classifier (SVM), Gradient Boost Classifier (GBC), and Gaussian Naive Bayes Classifier (GNB).

3.2.1 Supervised Learning

Supervised Learning is a subcategory of machine learning where the learning of an algorithm is supervised. This essentially means that the algorithm is taught by using examples with the output/target class. As shown in Figure 3.1 below, the data collected are first labeled into its respective categories by a supervisor. Next, the data are pre-processed and split into train and test data sets. 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. For prediction, the supervised machine learning algorithms take the unseen real data and attempt to make a determination of its label by using patterns learned during the training process.

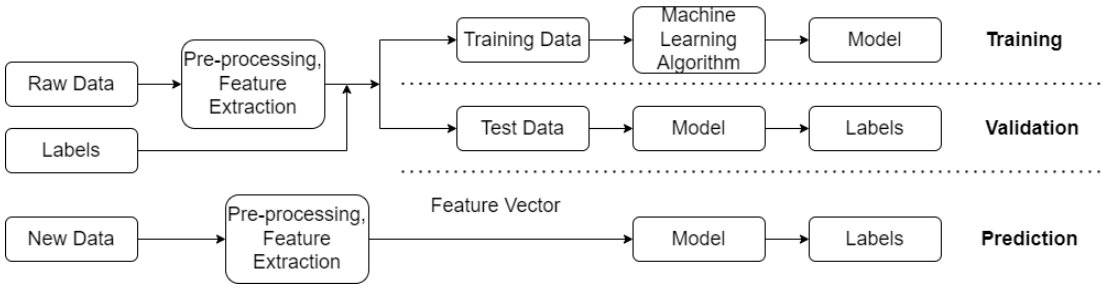

Figure 3.1: Supervised Learning

3.2.1.1 K-Nearest Neighbors (KNN) [6]

The KNN, a classification algorithm works by estimating the test data point in a group, based on its nearest "K" number of neighbors as shown in Figure 3.2 below. 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.

There are no predefined statistical methods to find the optimal value of K. First, initialize a random K value and start computing distances between test points and trained label points and update the distance metrics. Choosing a small value of K leads to unstable decision boundaries. The substantial K value is better for classification as it leads to smoothening the decision boundaries. Then, derive a plot between error rate and K denoting values in a defined range and choose the K value as having a minimum error rate.

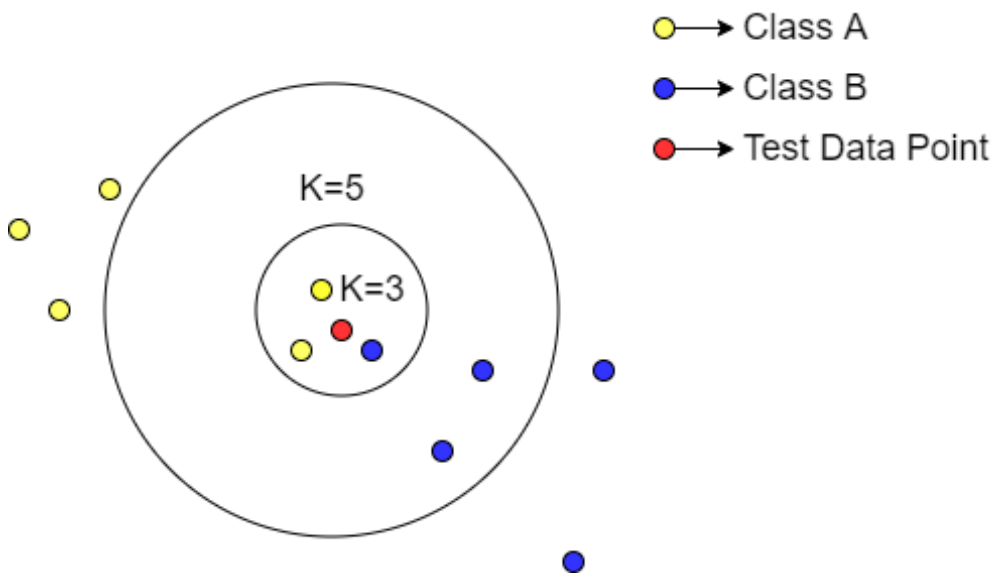

Figure 3.2: K-Nearest Neighbors Classifier

3.2.1.2 Random Forest Classifier (RFC) [9]

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 model's final prediction as shown in Figure 3.3 below.

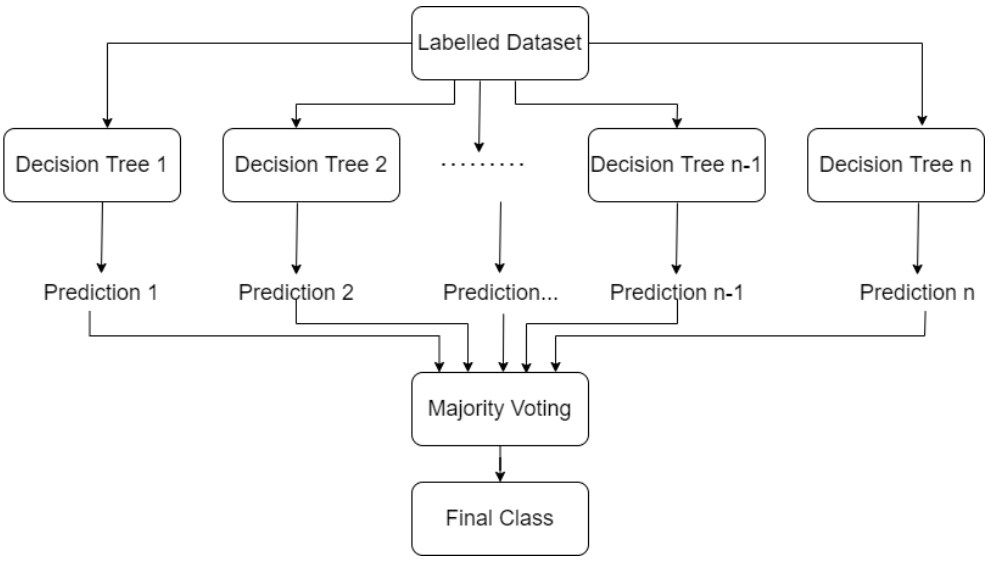

Figure 3.3: Random Forest Classifier

3.2.1.3 Support Vector Machine Classifier (SVM) [10]

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 differentiate 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 as shown in Figure 3.4 below.

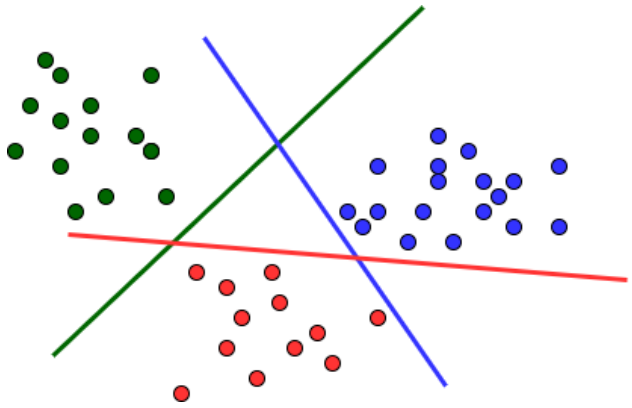

Figure 3.4: Support Vector Machine Classifier

3.2.1.4 Gradient Boost Classifier (GBC) [5]

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 as shown in Figure 3.5 below. The process effectively reduces the error over several iterations, which helps the algorithm to provide its final predictions.

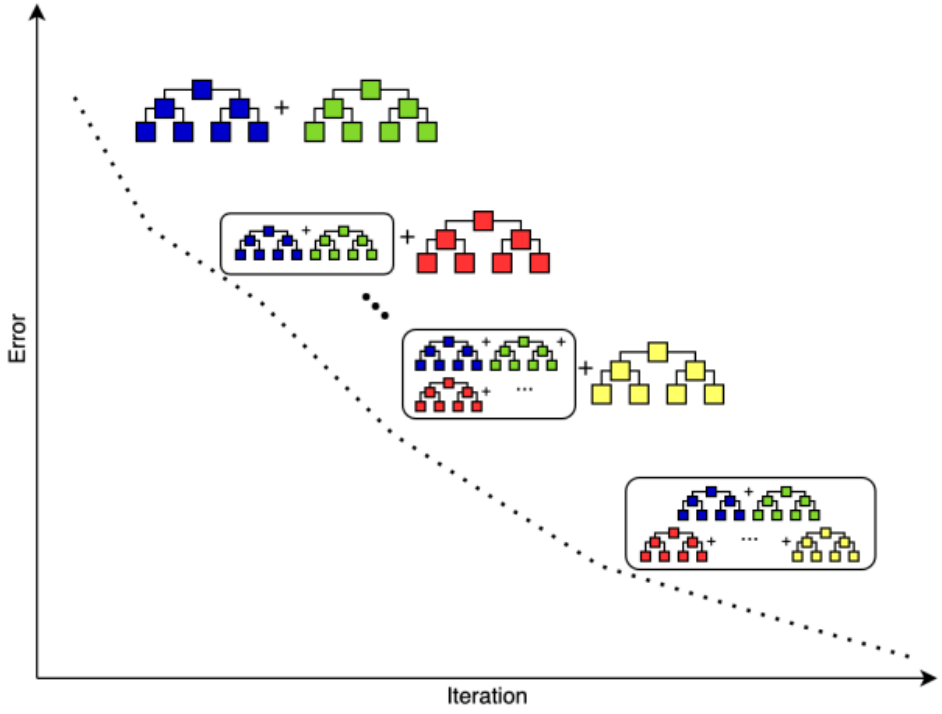

Figure 3.5: Gradient Boost Classifier

3.2.1.5 Gaussian Naive Bayes Classifier (GNB) [4]

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 (equation 3.1 below) which helps define the probability of the occurrence of hypothesis A after the data B, is already given.

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

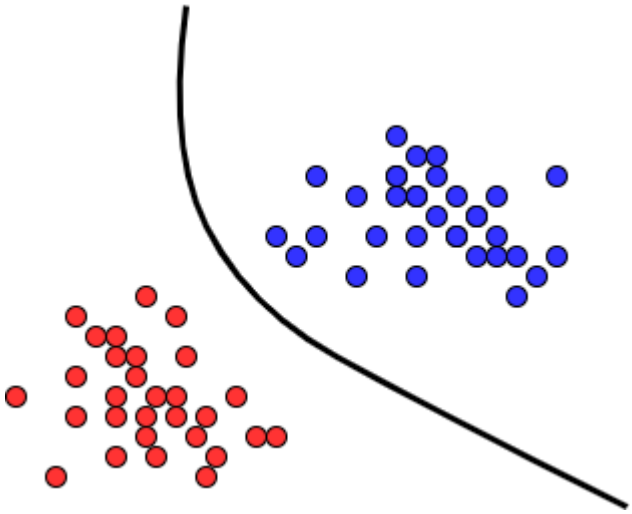

Figure 3.6: Simple Gaussian Naive Bayes Classifier

3.3 Model performance metrics

The metrics used to analyze each model performance were the F1 score and the accuracy score.

3.3.1 F1 Score [2]

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 misclassifications made.

Precision, also known as the positive predictive 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 equation

3.2 below:

$$\text{Precision} = \frac{\text{Number of True Positives}}{\text{Number of True Positives + Number of False Positives}} \tag{3.2}$$

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 equation 3.3 below:

$$\text{Recall} = \frac{\text{Number of True Positives}}{\text{Number of True Positives + Number of False Negatives}} \tag{3.3}$$

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 equation 3.4 below:

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision + Recall}} \tag{3.4}$$

3.3.2 Accuracy Score [1]

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 as shown in the equation below. Additionally, the percentage of this score can be calculated to obtain the accuracy of a classifier in terms of a percentage. The formula for accuracy score is given by equation 3.5 below.

$$\text{Accuracy Score} = \frac{\text{Number of correct predictions}}{\text{Number of total predictions}} \tag{3.5}$$

3.4 Experimentation Setup and Data Collection

3.4.1 Data collection environment

We create a data collection environment as shown in Figure 3.7. We configure a Wi-Fi router so that it can observe and record traffic between users' IoMT devices on the local network and the rest of the Internet. All traffic between Wi-Fi devices on the LAN or from IoMT devices to the Internet traverses this router. We connect all IoMT devices to a Wi-Fi router via Smartphone and use Wireshark to capture all traffic traversing the router into a stream of packets and save the packets in Packet Capture (PCAP) files for offline analysis.

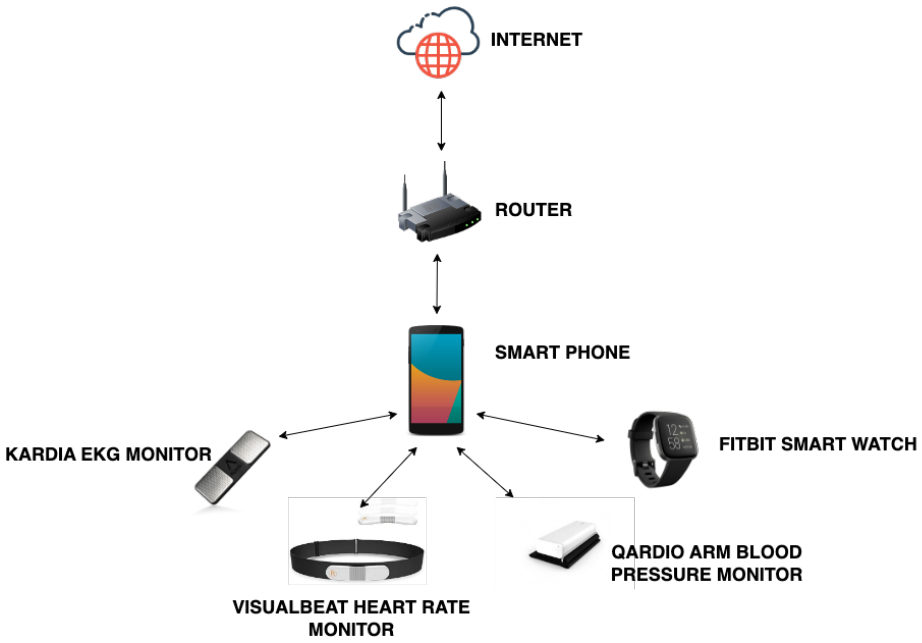

Figure 3.7: Data collection environment

We used four commercially available popular IoMT devices in our experimentation setup covering a range of device types, manufacturers, and privacy concerns.

- FitBit Smartwatch [3]
- QardioArm Smart Blood Pressure Monitor [8]

- Kardia EKG Monitor [7]
- VisualBeat Heart Rate Monitor [11]

The FitBit Smartwatch has different uses like measurement of steps, calories burned, distance traveled, sleep tracking, heart rate monitoring, workout mode, fitness tracking etc. QardioArm Smart Blood Pressure Monitor is a device used for measuring blood pressure. It also keeps track of a patient's blood pressure history and generates reports according to user requests. Kardia EKG Monitor records medical-grade heart rhythm data in seconds and is capable of detecting up to six of the most common arrhythmias namely Atrial Fibrillation, Bradycardia, Tachycardia, Sinus Rhythm with PVCs (Premature ventricular contractions), Sinus Rhythm with SVE (Supraventricular Ectopy) and Sinus Rhythm with Wide QRS. It is also capable of maintaining heartbeat history and generating reports for patients. The VisualBeat Heart Rate Monitor provides continuous real-time heart rate tracking for up to 24 hours. These 4 devices are by no means exhaustive of the wide range of available IoMT products, however, they encompass a variety of device types from large and small manufacturers.

3.4.2 Separate traffic into per-device packet streams from overall network traffic

3.4.2.1 Separate entire traffic into packet streams

An adversary must first divide recorded network traffic into meaningful streams that can be used for further traffic analysis. In general use cases, the home gateway router acts as a Network Address Translator (NAT), rewriting local Internet Protocol (IP) addresses of individual devices connected to a router to a single public IP address given to the router by the Internet Service Provider(ISP). This prevents an adversary from using IP addresses to divide traffic into per-device packet sets. However, it is always possible to separate network traffic into streams by the external IP address of the server communicating with the IoMT devices ("service IP") and, in cases where multiple devices use the same service IP, the TCP port rewritten by the NAT can be used to divide traffic into

per-device packet sets. While the devices we studied communicate with multiple separate service IPs, we discovered that the adversary typically only needs to identify a single packet representing a particular IoMT device to separate entire traffic into a group of packet streams.

3.4.2.2 Label packet streams by IoMT device type

Once individual streams have been separated, the adversary next identifies what IoMT device most likely is responsible for each stream. For this, first we get the IP address of the destination server to which each IoMT device connects to and map that address to the domain name to get device name or its manufacturer. Then, we label all the packet streams with corresponding IoMT devices and saved them to a per-device PCAP file. In our case studies, the Domain Name System (DNS) queries associated with each stream could be mapped to a particular device, as described later in section 4.2.1. However, multiple devices from the same manufacturer might communicate with the same service IPs, making device identification using DNS more difficult. For example, the Karida EKG Monitor queried domains that could have been used by any type of AliveCor device. Finding the solutions to this problem will be the subject of further research/study.

CHAPTER 4
DEVICE AND ACTIVITY IDENTIFICATION USING MACHINE LEARNING CLASSIFIERS

To conduct the research on machine learning based privacy attack, we first look at the possible attacking techniques that can be used to identify the users' IoMT devices. Specifically, we look at the methods used to automatically identify IoMT devices based on the DNS query and metadata involved with encrypted data transfer that are commonly seen on the IoMT devices that we are investigating. It is worth to note that those DNS query or metadata could also be easily obtained by the attackers just by capturing traffic into PCAP files. Different from most current works in this field that just uses a single machine learning classifier, we aim to investigate the effectiveness of a variety of machine learning classifiers for automatic device identification. Those classifiers include K-Nearest Neighbors, Support Vector Machines, Random Forest, Gradient Boost Classifier and Gaussian Naive Bayes Classifier. We compare the performance of all classifiers in terms of F1 score and accuracy, and propose the best classifier having highest accuracy and performance.

4.1 Training dataset preparation from the PCAP files

A PCAP file containing n number of packets

$$p_1, p_2, p_3, ..., p_n$$

is created for each IoMT device as described in section 3.4.2.2. From PCAP file of each IoMT device, we extracted 13 features listed in Table 4.1 using tshark command. These features include protocols used in different network layers, source and destination ports, packet size and window size. The following tshark command is used to extract features from a pcap file:

tshark -r pcapfile.pcapng -E header=y -E separator=, -T fields -e frame.len -e tcp.srcport -e tcp.dstport -e tcp.window_size -e ip.proto -e ip.opt.padding -e ip.opt.ra -e arp -e icmpv6 -e http -e ssdp -e dns -e mdns > features.csv

This command extracts all the features provided with -e attributes from pcapfile.pcapng and creates a new file features.csv output file with features as column names.

Type	**Features**
Link Layer Protocol	ARP
Network Layer Protocol	IP, ICMPv6
Transport Layer	Window Size
Application Layer Protocol	HTTP, DNS, MDNS, SSDP
IP Options	Padding, RouterAlert
Packet Content	Size
Port class	Source, Destination

Table 4.1: Application layer and corresponding protocol, feature used

None of these features rely on packet payload, ensuring that fingerprints can be extracted from the header of an encrypted traffic. These 13 features were chosen because they are typically used during device association over Wi-Fi connection. Each packet was represented with vector representation

$$p_i = \{f_{1,i}, f_{2,i}, f_{3,i}, ..., f_{13,i}\}$$

where i $\in$ {1,...,n}. The set of features we extracted from the PCAP files had several features that were not useful for device fingerprinting so, we chose to consider only the important features as removing unnecessary features would improve efficiency in a real world scenario by reducing the memory, processing power, time required for model training. Therefore, we calculated the important scores for the features by using the Random Forest. The reason for selecting Random Forest is that it has built-in feature selection based on its importance. It calculates feature importance using Gini importance and Mean Decrease Accuracy in each decision tree and the final feature importance would be the average of all decision tree feature importance. We calculated the feature importance score using traffic data of all 4 IoMT devices and is shown in Figure 4.1 below.

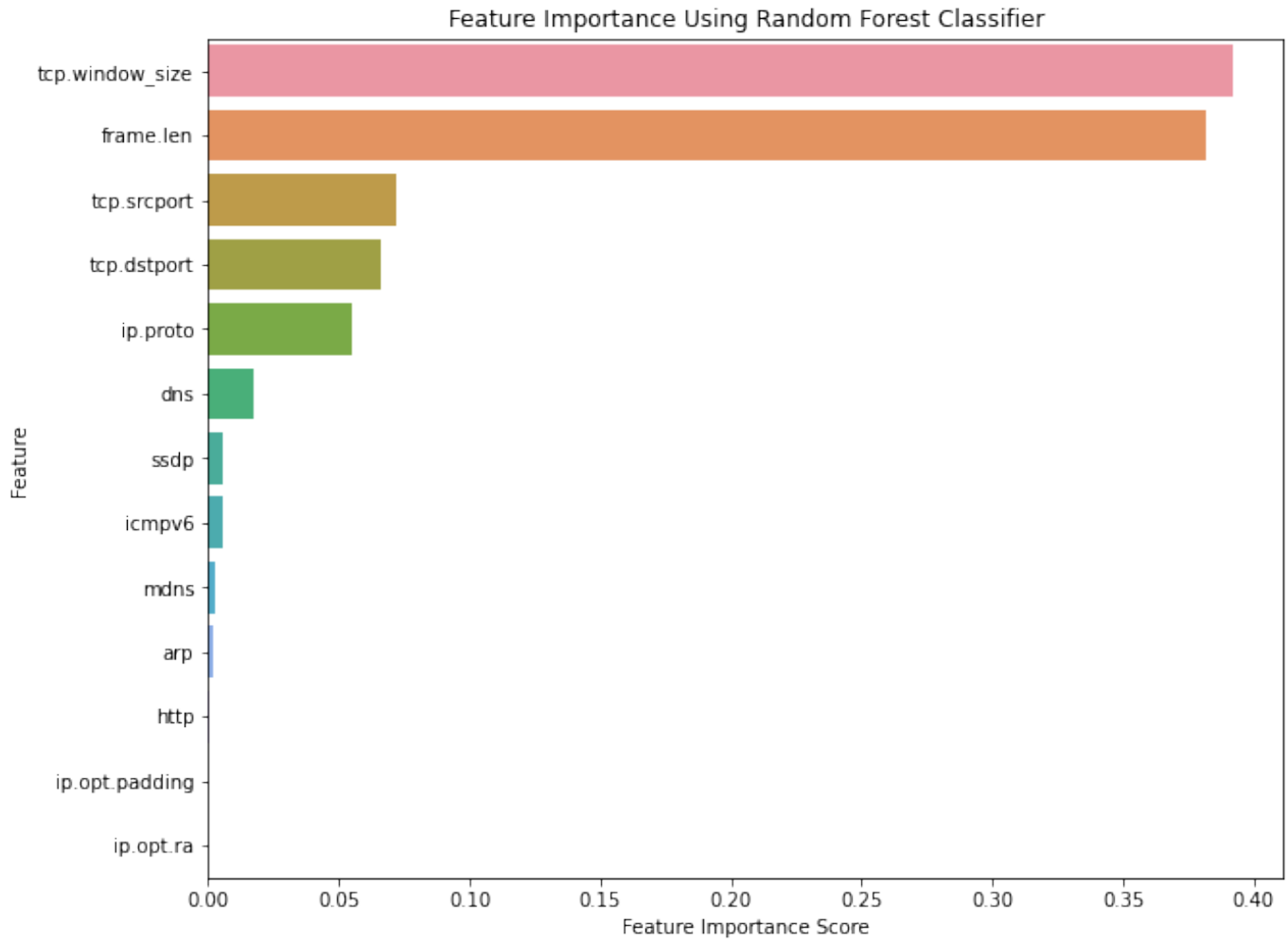

Figure 4.1: Feature importance scores using Random Forest Classifier

We set a threshold of 0.05 for the importance score of the feature and chose 5 features namely tcp.window_size, frame.len, tcp.srcport, tcp.dstport and ip.proto for which the importance score was greater than 0.05 and discarded 8 features namely dns, ssdp, icmpv6, mdns, arp, http, ip.opt.padding and ip.opt.ra for which the importance score less than 0.05. These features were removed from the dataset because they did not provide any valuable contribution or negatively affected the runtime, CPU and memory usage or significantly reduced the accuracy of our classifier models. 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.

Hence, a device fingerprint is a 5*n matrix F with each column representing a packet received

with order i $\in$ {1,...,n} and each row represents a packet feature as shown in the equation 4.1 below.

$$F = \begin{pmatrix} f_{1,1} & f_{1,2} & f_{1,3} & \cdots & f_{1,n} \\ f_{2,1} & f_{2,2} & f_{2,3} & \cdots & f_{2,n} \\ f_{3,1} & f_{3,2} & f_{3,3} & \cdots & f_{3,n} \\ f_{4,1} & f_{4,2} & f_{4,3} & \cdots & f_{4,n} \\ f_{5,1} & f_{5,2} & f_{5,3} & \cdots & f_{5,n} \end{pmatrix} \tag{4.1}$$

4.2 IoMT device identification/fingerprinting

There are several methods to perform device identification/fingerprinting, depending on the information available to the network observer.

4.2.1 Using DNS Queries

For 3 tested IoMT devices except VisualBeat Heart Rate Monitor, DNS query was sufficient for identifying the devices user owns by mapping the DNS Query to device name as shown in Table 4.2 below. Though all the devices use encryption for encrypting the data during transmission, after being able to successfully classify/identify IoT devices, we can say encryption alone is not enough to prevent privacy vulnerabilities. The following figures 4.2, 4.3 and 4.4 show the IoMT devices and corresponding DNS queries made by them during a representative packet capture, which can be easily mapped to a specific device or manufacturer.

Device Name	DNS Query
FitBit Smartwatch	fsc.fitbit.com
Kardia EKG Monitor	us-kardia-production.alivecor.com
QardioArm Smart Blood Pressure Monitor	api.getqardio.com
VisualBeat Heart Rate Monitor	epdg.epc.mnc260.mcc310.pub.3gppnetwork.org

Table 4.2: IoMT device and corresponding DNS Query

4.2.1.1 FitBit Smartwatch

DNS Query: fsc.fitbit.com

```
> Frame 429425: 94 bytes on wire (752 bits), 94 bytes captured (752 bits) on interface -, id 0
> Ethernet II, Src: f2:99:7e:d7:90:b3 (f2:99:7e:d7:90:b3), Dst: BelkinIn_ad:66:7a (c4:41:1e:ad:66:7a)
> Internet Protocol Version 6, Src: fd88:7f11:3fb3:0:f9c6:e7d3:1194:46cf, Dst: fd88:7f11:3fb3::1
> User Datagram Protocol, Src Port: 63718, Dst Port: 53
v Domain Name System (query)
    Transaction ID: 0xd4f8
  > Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  v Queries
    v fsc.fitbit.com: type A, class IN
        Name: fsc.fitbit.com
        [Name Length: 14]
        [Label Count: 3]
        Type: A (Host Address) (1)
        Class: IN (0x0001)
    [Response In: 429452]
```

Figure 4.2: FitBit DNS Query

4.2.1.2 Kardia EKG Monitor

DNS Query: us-kardia-production.alivecor.com

```
> Frame 43: 113 bytes on wire (904 bits), 113 bytes captured (904 bits) on interface -, id 0
> Ethernet II, Src: da:19:c8:0b:bf:30 (da:19:c8:0b:bf:30), Dst: BelkinIn_ad:66:7a (c4:41:1e:ad:66:7a)
> Internet Protocol Version 6, Src: fd88:7f11:3fb3:0:d85c:d162:f258:47aa, Dst: fd88:7f11:3fb3::1
> User Datagram Protocol, Src Port: 45502, Dst Port: 53
v Domain Name System (query)
    Transaction ID: 0xdc5e
  > Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  v Queries
    v us-kardia-production.alivecor.com: type A, class IN
        Name: us-kardia-production.alivecor.com
        [Name Length: 33]
        [Label Count: 3]
        Type: A (Host Address) (1)
        Class: IN (0x0001)
    [Response In: 44]
```

Figure 4.3: Kardia DNS Query

4.2.1.3 QardioArm Smart Blood Pressure Monitor

DNS Query: api.getqardio.com

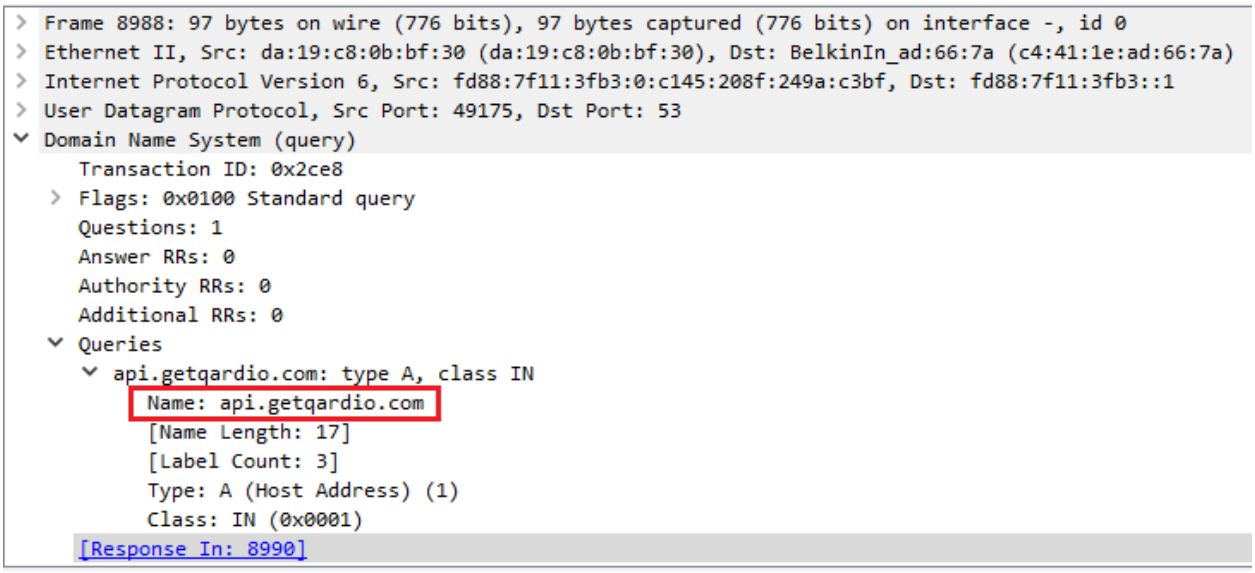

Figure 4.4: QardioArm DNS Query

An adversary could use a laboratory setup like our own to capture network traffic. Network observers can easily identify the user IoMT devices from the DNS query itself. For instance, from the dns query iphone-cdn-client.fitbit.com or iphone-cdn-client.fitbit.com.cdn.cloudflare.net one can easily identify the device being used is FitBit smartwatch.

4.2.2 Using Machine Learning on traffic metadata

We cannot always guarantee that a DNS query contains the information or keywords that can provide hints about the server. Among 4 IoMT devices we tested, the DNS query of Visualbeat Heart Rate Monitor did not contain any keyword that could provide some hint about the cloud server as shown in Figure 4.5 below.

In such scenario, we can use machine learning technique on traffic metadata to identify the device on the network. Device fingerprinting using machine learning is based on passively observed network traffic metadata, especially packet header information, as payloads are completely

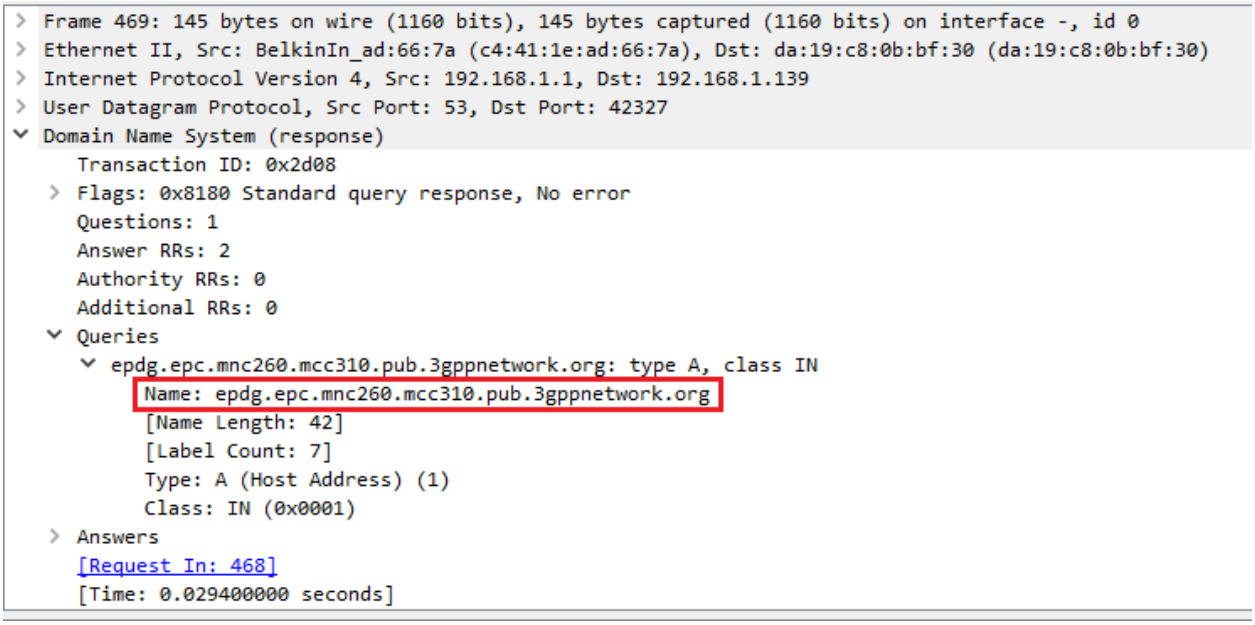

Figure 4.5: VisualBeat DNS Query

encrypted these days. Each device on the network is characterized by a distinguishable sequence of communications initiated by that device, which our machine learning algorithms for fingerprinting attempt to capture.

4.2.2.1 Feature Selection

For device fingerprinting, we considered five features namely tcp.window_size, frame.len, tcp.srcport, tcp.dstport and ip.proto to train our device fingerprinting classifiers.

4.2.2.2 Data Pre-processing

In order to make the data set ready to train and test for machine learning classifiers, we preprocessed it using several techniques. The following are the preprocessing techniques used in this research:

1. **Data Cleaning and Splitting**

   Feature values for link, network, transport and application layers were text values, for instance, if a packet is using Internet Protocol then the value for ip.proto column in features.csv

will be Internet Protocol, else empty value. In such cases, for any feature having the text values were replaced by 1 and empty values by 0. Such types of binary features are set to 1 if selected communication protocols are used, otherwise 0. Then each row is labeled with the corresponding target class, i.e. target IoMT device. Each IoMT device is assigned a unique identifier as target class, as shown in Table 4.3 below.

Then the data in the csv file was loaded into a data frame and the labels were separated out of the data frame by 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.

Device Name	**Class**
FitBit Smartwatch	1
Kardia EKG Monitor	2
QardioArm Smart Blood Pressure Monitor	3
VisualBeat Heart Rate Monitor	4

Table 4.3: IoMT device and corresponding class

2. Standardizing Features

Standardizing features is a requirement for many classifiers to achieve high accuracy. Features like frame length and window size have a wide range of values starting from 0 to a hundred thousands. Being a vast difference in the range and KNN with a Euclidean distance measure is sensitive to magnitudes, the machine learning model makes the underlying assumption that higher ranging numbers have superiority of some sort, so these more significant numbers start playing a more decisive role while training the model. Hence, to avoid this scenario, it is always recommended to bring all features in the same standing. We did apply standardization so that one significant number didn't impact the model just because of their large magnitude. Standard Scaler was used for standardization, which scaled the data such that the distribution centered around 0, with a standard deviation of 1.

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 fillna method provided by the pandas data analysis tool. The two last features represent the source and destination ports used, if any, mapped to network port class 0, 1, 2 and 3. The purpose of mapping the source and destination ports used is to handle the wide range of network ports using few values.

- no port f = 0
- well-known port [0, 1023] f = 1
- registered port [1024, 49151] f = 2
- dynamic port [49152, 65535] f = 3

4.2.2.3 Training and Testing

Since only one packet information was not enough to identify the IoMT device, we aggregated network traffic statistics over a period of time and built a second fixed-size fingerprint F′, composed of the 12 first unique vector packets p from F, concatenated to produce a 60-dimensional feature vector (12 packets × 5 features). The procedure to produce device fingerprint matrix F from per-device PCAP file after separating each device traffic out of entire traffic generated from a single smart phone or Wi-Fi access point is described in sections 3.4.2 and 4.1 above.

$$F' = \{f_{1,1}, f_{2,1}, ..., f_{5,1}, f_{1,2}, f_{2,2}, ..., f_{5,2}, f_{1,3}, ..., f_{5,3}, ..., f_{1,i}, f_{2,i}, ...f_{4,i}, f_{5,i}\}$$

We did the same analysis with other numbers of packets like 5, 8, 10, 12 and 15, but we found that 12 packets were a good trade-off for F′ length and was long enough to distinguish device-types and short enough to be fully filled with unique packets from F.

After pre-processing the dataset, 80% of it was used to train our five classifier models. The training process was performed using the fit method provided by sklearn, which creates the model

by fitting the training dataset. In the following section we have evaluated the performance of all 5 classifiers based on the predictions made by each model.

4.2.2.4 Classifier Performance

We calculated the F1 score using the evaluation metric library provided by sklearn.metrics [2]. The highest F1 score was provided by the KNN classifier for all tested IoMT devices, which is plotted below in Figure 4.6 along with the F1 score for all classifiers and IoMT devices, where it can be further seen that the KNN model outperforms other models for device fingerprinting.

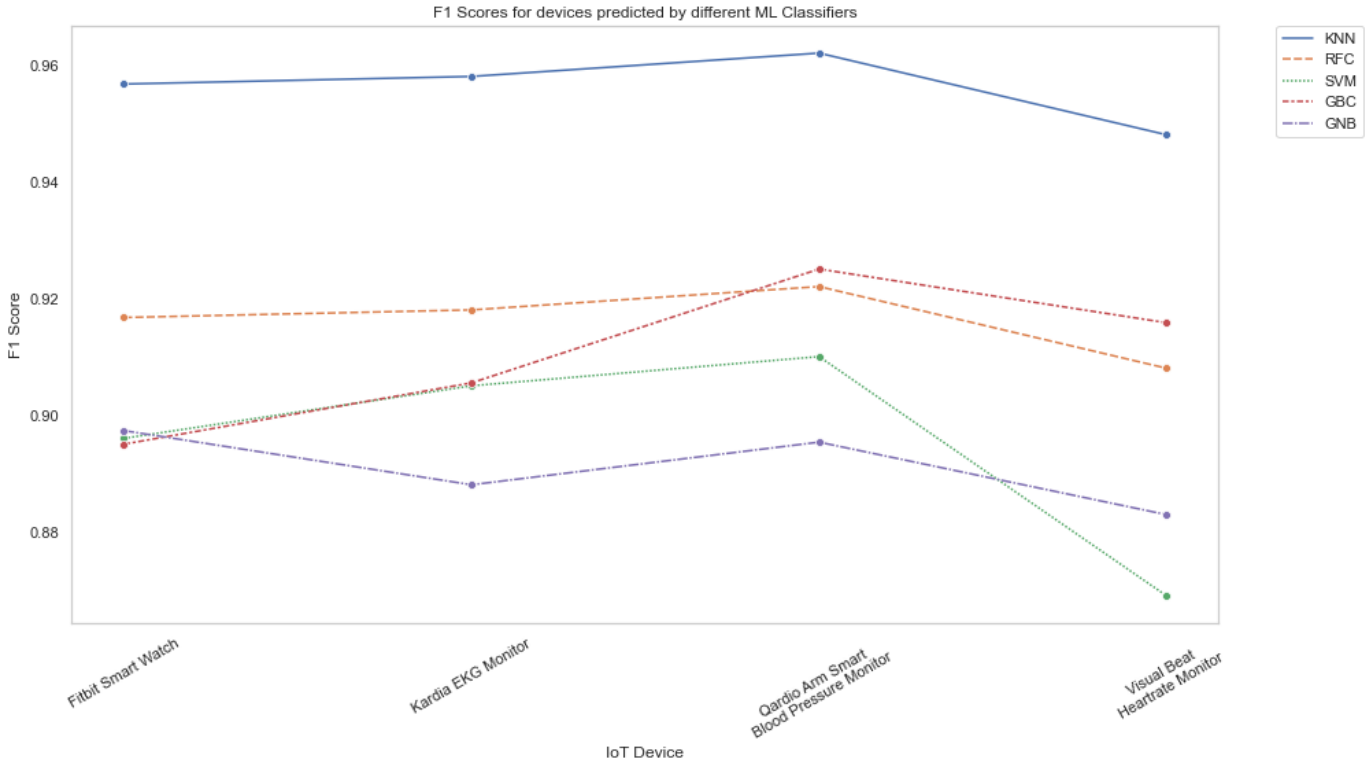

Figure 4.6: F1 scores of device identification for GBC, GNB, KNN, RFC and SVM

The function to calculate the accuracy of our machine learning models used to perform multi-class classification was provided by the sklearn.metrics library [1]. After making the predictions on the test data set, we found that the KNN classifier was the most accurate model for device fingerprinting with an accuracy of 94.6%. The second most accurate classifier was the Random Forest Classifier (RFC) with an accuracy of 91.9%, then the Gradient Boost Classifier (GBC) with

an accuracy of 89%, Support Vector Machine (SVM) classifier with an accuracy 83.8%, and finally the Gaussian Naive Bayes Classifier (GNB) with an accuracy of 80.1% as shown in the Figure 4.7 below.

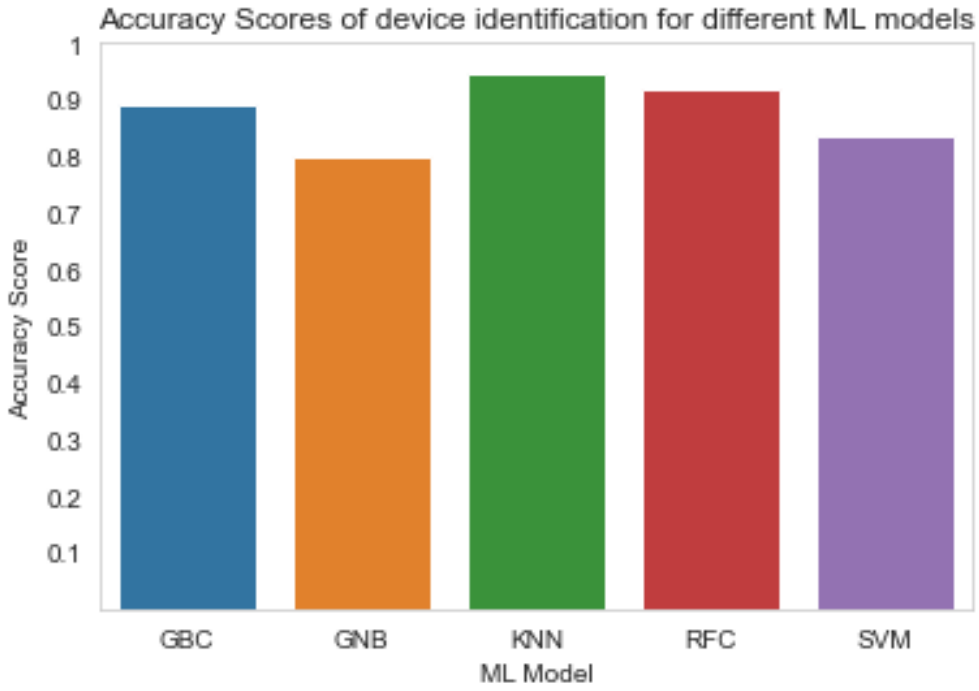

Figure 4.7: Accuracy of device identification for different ML models

As shown in the figure 4.5 above, the domain name of the VisualBeat Heart Rate Monitor server is epdg.epc.mnc260.mcc310.pub.3gppnetwork.org. By just simply viewing this domain name, we were unable to tell that the device connecting to this server is VisualBeat since, unlike other domain names, this did not have any hint/keyword from which VisualBeat can be inferred. However, KNN, one of our trained machine learning classifiers correctly identified that device as VisualBeat Heart Rate Monitor with accuracy of 94.6%. This shows that machine learning classifiers can be used to identify the device even when it is impossible to infer using the DNS query or domain name.

4.3 IoMT event/activity identification

4.3.1 Examine traffic rates

Once an adversary identifies packet streams for a particular device, one or more of the streams are likely to encode activity performed on the device. When an interaction between the device and the user occurs, a significant amount of data is transmitted, which leads to a significant increase in the traffic rate. After this data exchange, the data transmission drops to the minimum until a new interaction starts. When there is no activity, only the minimum amount of continuation packets like heartbeat messages are sent to minimize the device's power and bandwidth consumption. We also observed that almost the same amount of data transfer occurs for the same activities. All this information allows us to detect transitions between the activities or events of the device. So, simply plotting send/receive rates of the streams (bytes per second) revealed potentially private user interactions for each device we tested. An adversary with a laboratory of IoMT devices can easily correlate variations in traffic rates with known user interactions. They can then map similar variations from live traffic to user activities. Even without a laboratory of IoMT devices, an adversary can still infer user interactions from traffic variations if they have identified the device and know its limited purpose. For example, the QardioArm Smart Blood Pressure Monitor was both easily identified from DNS queries (api.getqardio.com), and has a limited purpose of blood pressure measurement. A traffic spike from the monitor at any time likely corresponds to when the user measured the blood pressure and lower traffic rate corresponds to idle state.

In our case, 3 out of 4 tested IoMT devices have only two states/events namely idle and measurement. For these 3 (Kardia EKG Monitor, QardioArm Smart Blood Pressure Monitor and VisualBeat Heart Rate Monitor) IoMT devices, we can easily identify their state/event (e.g., deciding if it is ON or OFF, idle or measuring/using, ACTIVE or INACTIVE) by observing the traffic rates; traffic spike infers device is in measurement state or currently in use and lower traffic rate infers idle state as shown with Kardia EKG Monitor traffic in Figure 4.8 below.

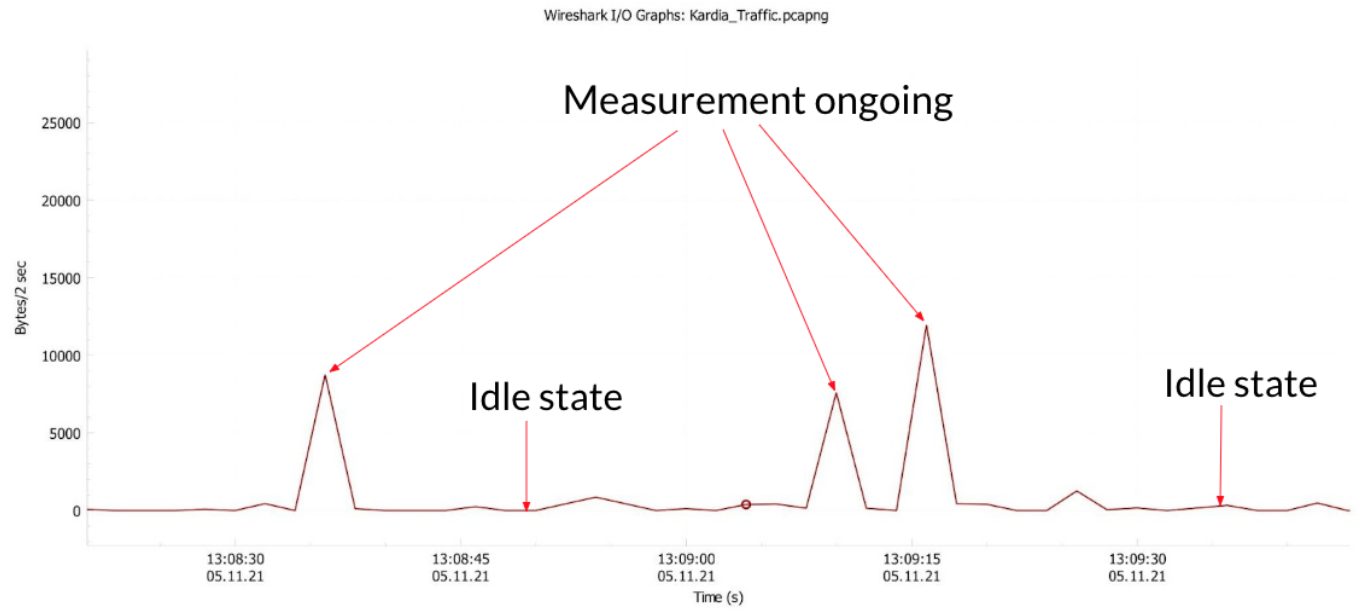

Figure 4.8: Kardia EKG Monitor Traffic when device events changed

However, for some IoMT device two or more activities or events may have a similar traffic rates/patterns which might be difficult for us to differentiate them by simply looking at the device's traffic rates. For instance, in Fitbit Smartwatch workout and running both cause the traffic rate to spike which could be difficult to distinguish between these two events just by seeing the traffic rate as shown in Figure 4.9 below.

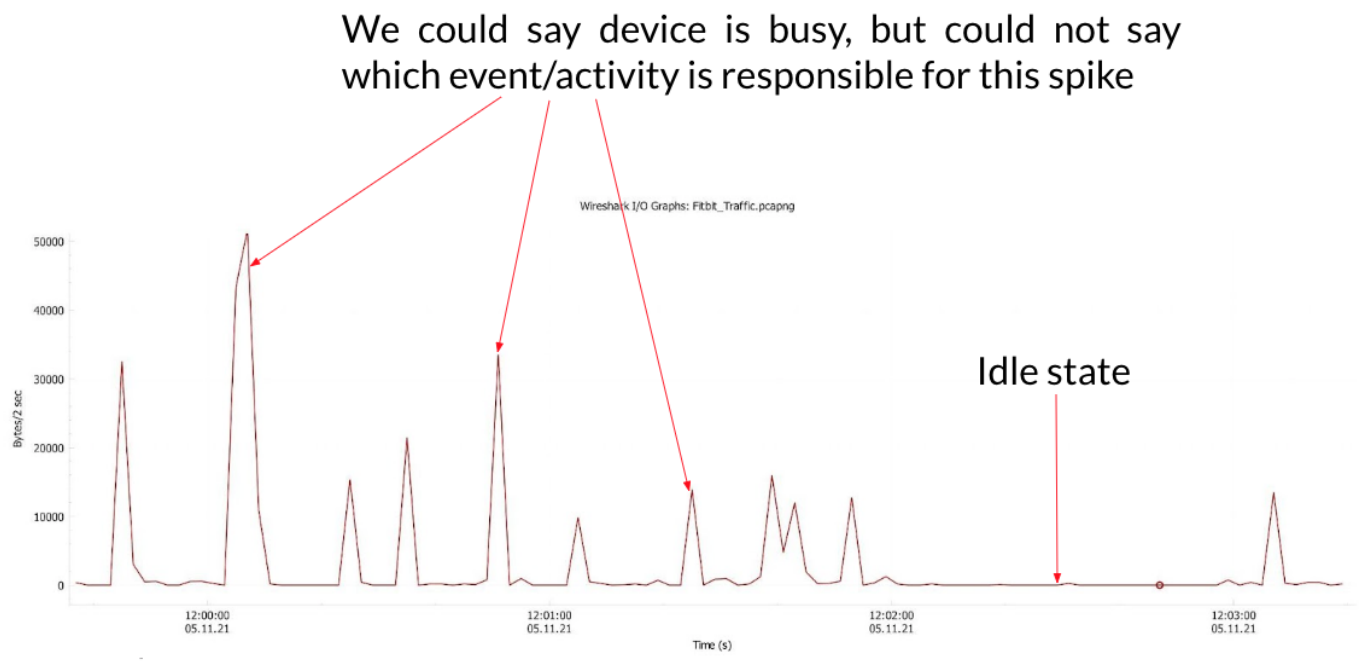

Figure 4.9: Fitbit Smartwatch Traffic when device events changed

In order to experiment if it is possible to differentiate between such events/activities of an IoMT device, we employed machine learning techniques as detailed in the next subsection.

4.3.2 Using Machine Learning on traffic metadata

As described in section 4.3.1, by simply plotting the traffic rate, we can easily identify the event of the device having only two distinct events like ON or OFF, ACTIVE or INACTIVE among which one causes traffic rate to spike and other causes to fall. However, the same approach does not work for the devices having more than one event causing traffic rate to either spike or fall, in other words events having similar traffic rates/patterns. In such scenarios, applying machine learning on traffic metadata can correctly identify the devices events or user activities.

4.3.2.1 Feature Selection

For device event or user activity identification, we utilized the same training data set prepared above in section 4.1. Unlike device fingerprinting, we considered only one feature i.e. frame

length among 5 available features since the device event/user activity can be identified from the frame length/packet size of the traffic as described in section 4.3.1. We trained our event/activity identification classifiers with statistical measures min, max, sum, mean and standard deviation of frame length.

4.3.2.2 Data Pre-processing

Similar to the device fingerprinting study, we also need to make pre-processing of the data so that we have the right input for activity/event identification algorithms.

1. **Data Cleaning and Splitting**

   Each row is labeled with the corresponding target class i.e. target user activity. Each user activity is assigned a unique identifier as target class. Table 4.4 below contains the list of Fitbit smartwatch activities/events with corresponding target class.

Device	Activity/Event	Class
FitBit Smartwatch	Idle	0
	Workout	1
	Sleeping	2
	Running	3

Table 4.4: Fitbit activities and their corresponding classes

2. **Standardizing Features**

   We performed standardization of features as described in the item 2 of section 4.2.2.2 above.

3. **Numerical Imputation**

   We use numerical imputation to assign a value to missing values in any feature. Any missing value is imputed to the median values of each individual feature.

4.3.2.3 Training and Testing

We produced a 60-dimensional feature vector (12 packets × 5 features) then used 8/2 split cross validation(i.e. Train on randomly selected 80% of the data and test on 20% remaining data and

repeat the process for a certain number of times to get the average metrics) as described in the section 4.2.2.3 above. As described in section 4.3.1, for 3 IoMT devices except the Fitbit Smartwatch, we could easily identify the user activity by simply observing traffic rates. So, to identify Fitbit user activities we developed five machine learning classifiers and trained them using the training data set. We have presented the performance of each model based on how accurately it classifies the Fitbit events/user activities below.

4.3.2.4 Classifier Performance

The highest F1 score was provided by the KNN classifier for Fitbit by outperforming the rest of the classifiers as shown in the Figure 4.10 below.

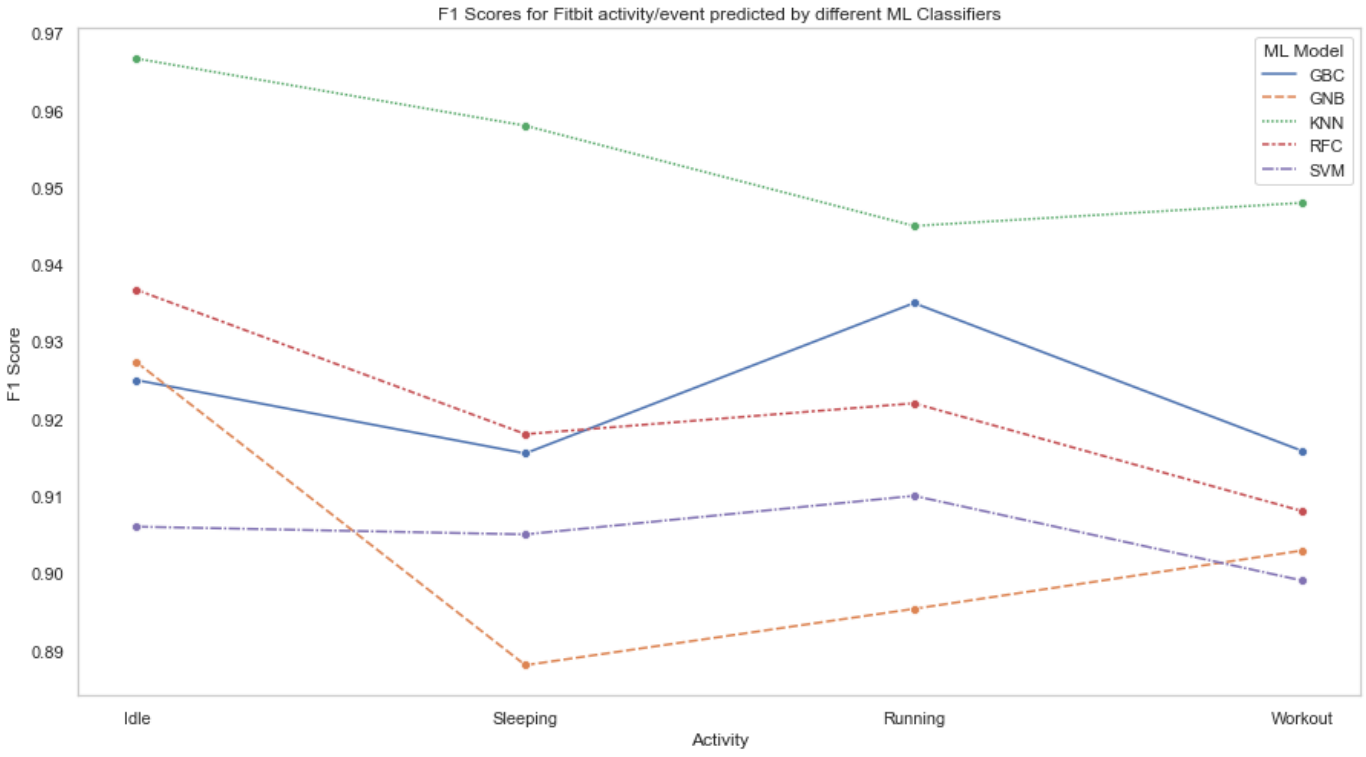

Figure 4.10: F1 scores of Fitbit activity/event classification for GBC, GNB, KNN, RFC and SVM

In the figure 4.11 below, we can clearly see that the KNN classifier is the most accurate model for the Fitbit event/activity classification with an accuracy of 93.1%. The second most accurate classifier is the Random Forest Classifier (RFC) with an accuracy of 92.8%, then the Support

Vector Machine (SVM) classifier with an accuracy 90%, Gradient Boost Classifier (GBC) with an accuracy of 88.7% and finally the Gaussian Naive Bayes Classifier (GNB) with an accuracy of 85.1%.

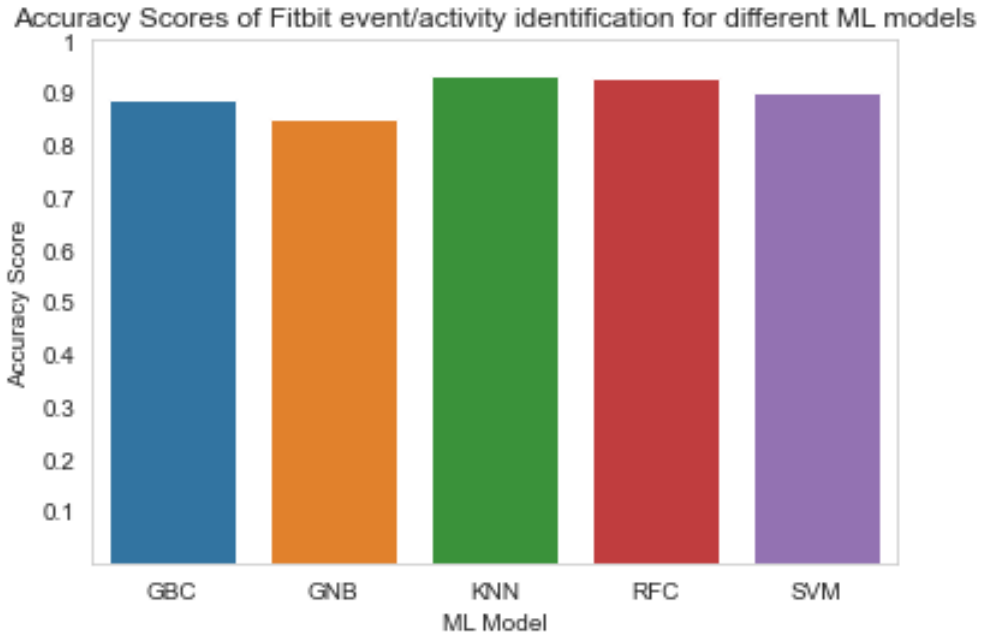

Figure 4.11: Accuracy of Fitbit activity/event classification for different ML models

We found that two activities, Workout and Running of Fitbit Smartwatch cause the traffic rate to spike creating the similar traffic pattern. Initially we tried to differentiate these two events by simply observing the traffic rates as described in section 4.3.1, but we could not exactly say which activity was responsible for that traffic spike. Either of the activity should be responsible, in that sense we could say the accuracy of traffic rate observation methodology was 50%. Also, with three distinct activities, accuracy of traffic rate observation methodology would be 33.33%, 25% with four distinct activities and so on. The accuracy of traffic rate observation methodology keeps decreasing with increase in number of distinct activities having same traffic rate/pattern. However, KNN, one of our trained machine learning classifiers correctly classified these two events with accuracy of 93.1%. Another important point is that the number of distinct activities does not affect the accuracy of machine learning classifiers at all. This is why machine learning classifiers come into the picture for the identification of such events/activities.

CHAPTER 5
DEFENSE AGAINST MACHINE LEARNING BASED PRIVACY ATTACKS

In the previous chapter, we successfully demonstrate that using machine learning classifiers we can identify the IoMT device and its events/activities. An adversary could use a laboratory setup like our own to identify the users' devices and activities performed on those devices. Knowing what devices a consumer owns can be a serious privacy violation by itself. For example, a consumer might not want an ISP knowing they own an IoMT blood sugar monitor or pacemaker. In fact, simply knowing a device a user owns could have unwanted advertising implications, or it enables other people to know one's potential health status and problems, medical history etc. To protect users' privacy that might be inferred from encrypted traffic metadata using machine learning, we propose the traffic shaping technique to make device and user activity identification more difficult for passive network observers.

5.1 Traffic Shaping

Both device and activity identification are the components of privacy attack using traffic metadata. Our main goal is to evade the machine learning classifiers to correctly identify the IoMT device and its event/activity with less classifier training time overhead and maintain the packet utility at the same time. To achieve this, we propose the technique to inject false traffic packets into the real traffic to alter the IoMT traffic pattern generated from IoMT devices. Each data set is a tuple of 5 features namely window size, IP protocol, source port, destination port and packet size (frame length). Traffic shaping is the way to mask the true rate of devices network traffic which involves shaping both upload and download traffic rates thereby exposing no information about devices and their corresponding events/activities. Hence, adversaries cannot differentiate different IoMT devices and device events or user activities.

Our traffic shaping algorithm first identifies the maximum packet length from entire traffic and then injects the false traffic having packet length equal to that maximum packet length in a fixed interval of every 5 seconds. We continued false traffic injection for a window interval that is enough to cover the real traffic as shown in Figures 5.1 and 5.2 below. We injected 10%, 20%, 30%, 40% and 100% false traffic from the real traffic by changing both window size and frame length(packet size) values when defending device identification, and only frame length values when defending event/activity classification to real traffic and re-trained all 5 classifiers with the combination of both false and real traffic. After making the predictions on the test data, we found that accuracy of all classifiers drastically decreased with the injection of false packets. Using traffic shaping, even if the user is not measuring blood pressure generating false activity for the blood pressure measurement will mask the user not measuring the blood pressure. An adversary cannot distinguish between real traffic and injected false traffic.

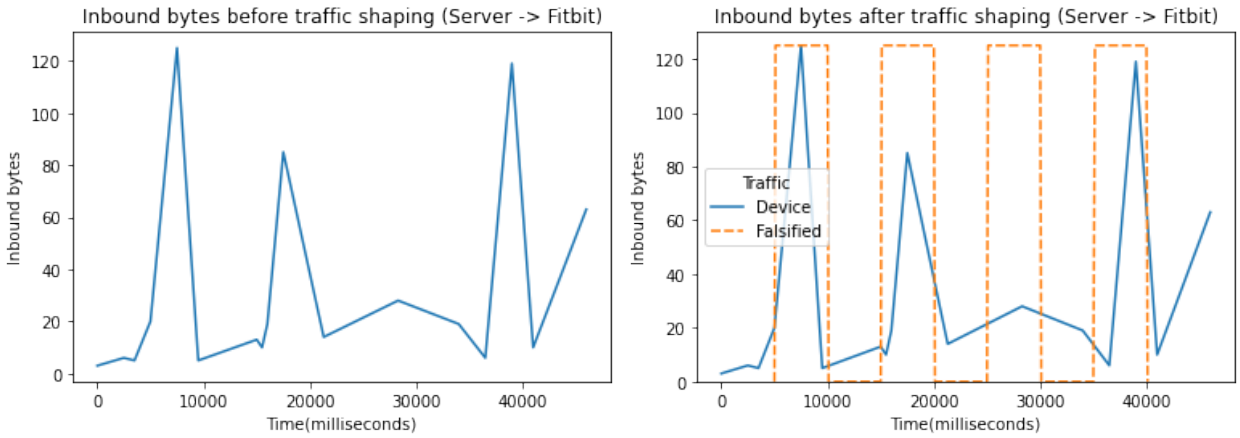

Figure 5.1: Inbound traffic traces of FitBit Smartwatch before and after traffic shaping

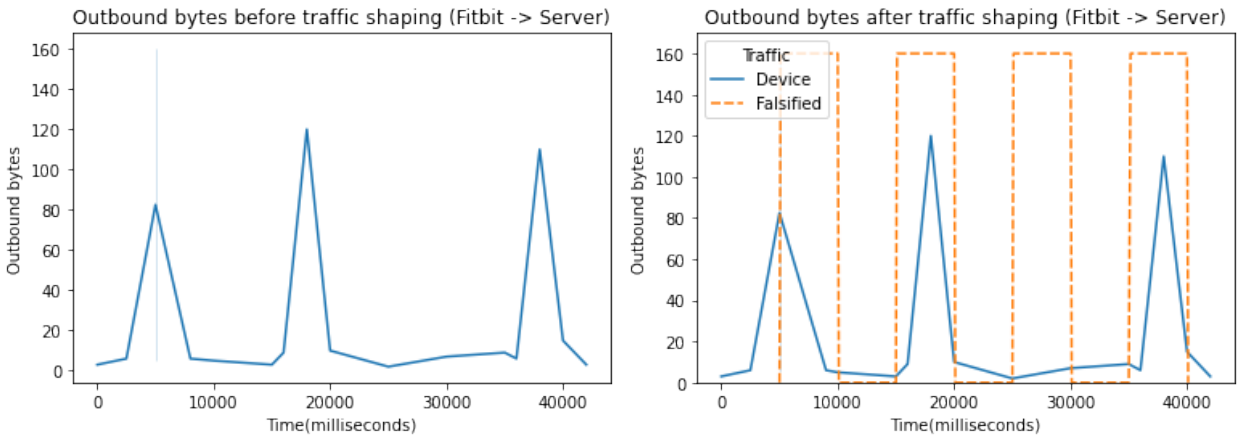

Figure 5.2: Outbound traffic traces of FitBit Smartwatch before and after traffic shaping

This is just a simulation of traffic shaping, bandwidth overhead is the main factor to consider while implementing real traffic shaping techniques. It is the ratio of network data sent with and without traffic shaping. For instance, a bandwidth overhead of 3 indicates that applying the traffic shaping technique results in 3 times as much traffic in bytes sent on the network than would be sent if the traffic were unshaped. A lower bandwidth overhead is always recommended because extra traffic contributes to network congestion and can consume more user data. Higher bandwidth results in higher network latency which most of the users do not tolerate nowadays.

5.2 Impact of Traffic Shaping on IoMT Device Identification

As shown in Figure 5.3 below, with 30% of false traffic injection, accuracy of the most accurate classifier - KNN, was decreased to 18.2% from 94.6%. The second most accurate classifier Random Forest Classifier (RFC) was reduced to 20% from 91.9%. This is due to the fact that randomly falsified data sets deteriorate traffic patterns used for device fingerprinting.

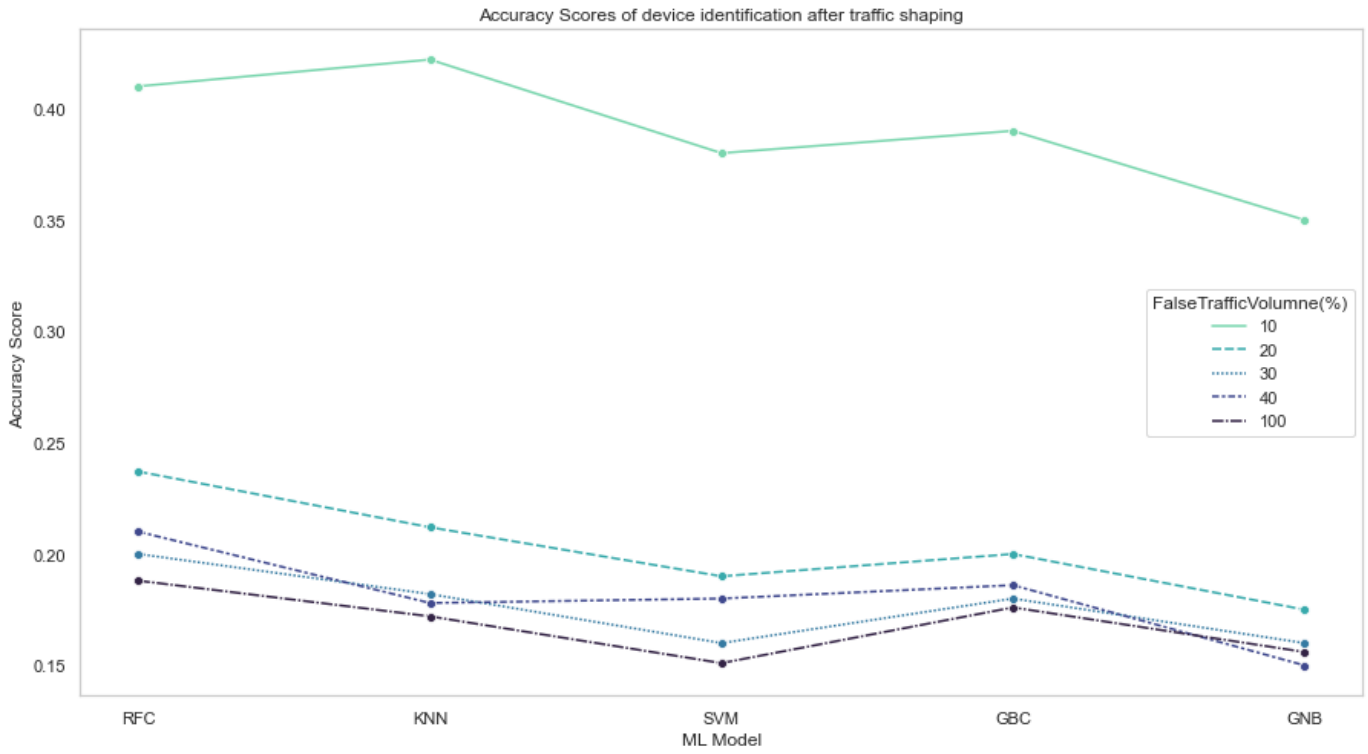

Figure 5.3: Accuracy of device fingerprinting after traffic shaping

As described in section 4.2.2.1 above, our device identification classifiers used five features namely window size, frame length, source port, destination port and IP protocol. Falsified data injected during traffic shaping changed values of two features window size and the frame length while other features values remain unchanged. The reason behind keeping ports and IP protocol values unchanged was to maintain the utility of each packet. Every application is bound to the specific port in the server which is the identity for the packet to be delivered to, if we alter these values the packets will not be delivered to the desired application in the server. The dramatically plunged accuracy of all the classifiers after injecting different percentages of false traffic from 0 to 30% demonstrate that traffic injection can be efficiently used for hiding the identity of devices from the adversary. However, we can also see that an even higher rate of false traffic does not help much. In fact, the effect of 100% false traffic does not differ much from 30% of false traffic.

We also plotted the average training time of device fingerprinting classifiers to study the effect of false traffic injection on their training time. The Figure 5.4 below shows the average training

time taken by device fingerprinting classifiers without and with 10% , 20%, 30%, 40% and 100% of false traffic injection to the original training data set. Training time for KNN increased by 1.8 times, by 1.5 times for RFC, SVM, GBC and GNB with addition of 30% false traffic.

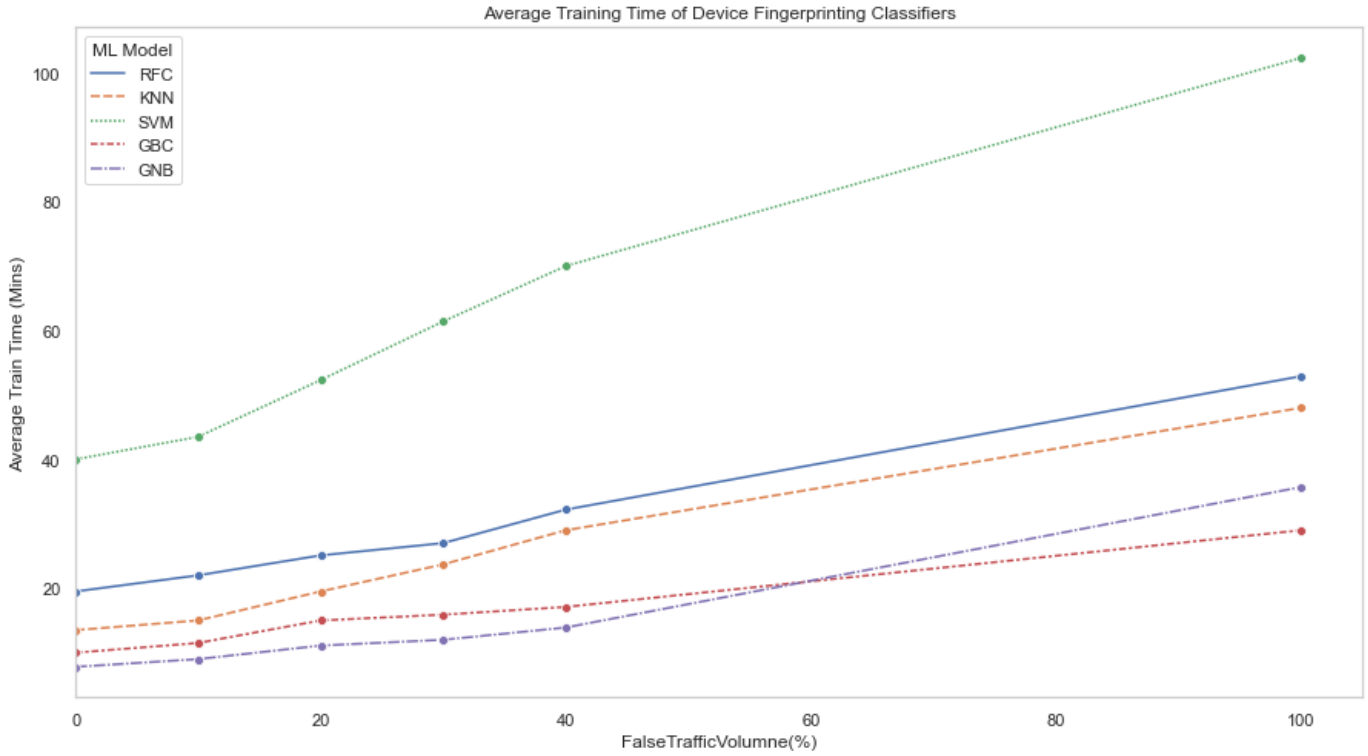

Figure 5.4: Average training time of device identification classifiers

5.3 Impact of Traffic Shaping on Activity/Event Classification

Similarly, for activity/event classification, adversaries leverage devices' known specific purposes to map changes in traffic rates to user activities. Most of the IoMT devices we used during our research have two states (measurement and idle), so if the traffic flow indicates a spike in traffic rate at a particular time, the adversary can infer that the user is performing measurement at that time and if the traffic flow indicates a fall in traffic rate, the adversary can infer that the device is idle at that particular time. As can be seen in the Figure 5.5 below, accuracy of all the classifiers plunges dramatically when injecting different percentages of false traffic volumes. With 40% of

false traffic injection, accuracy of the most accurate classifier, KNN was decreased to 32% from 93.1%, that of the second most accurate classifier Random Forest Classifier (RFC) was reduced to 28.5% from 92.8%. Besides significant decrease in accuracy of event/activity identification, injected fabricated traffic also make the inference of user activities from direct traffic rate plot infeasible. If we see the right side figure in the Figures 5.2 and 5.1, we could not infer any user activities from the change in traffic rate as entire traffic after traffic shaping are of fixed size (new traffic rate is indicated by orange dashed line).

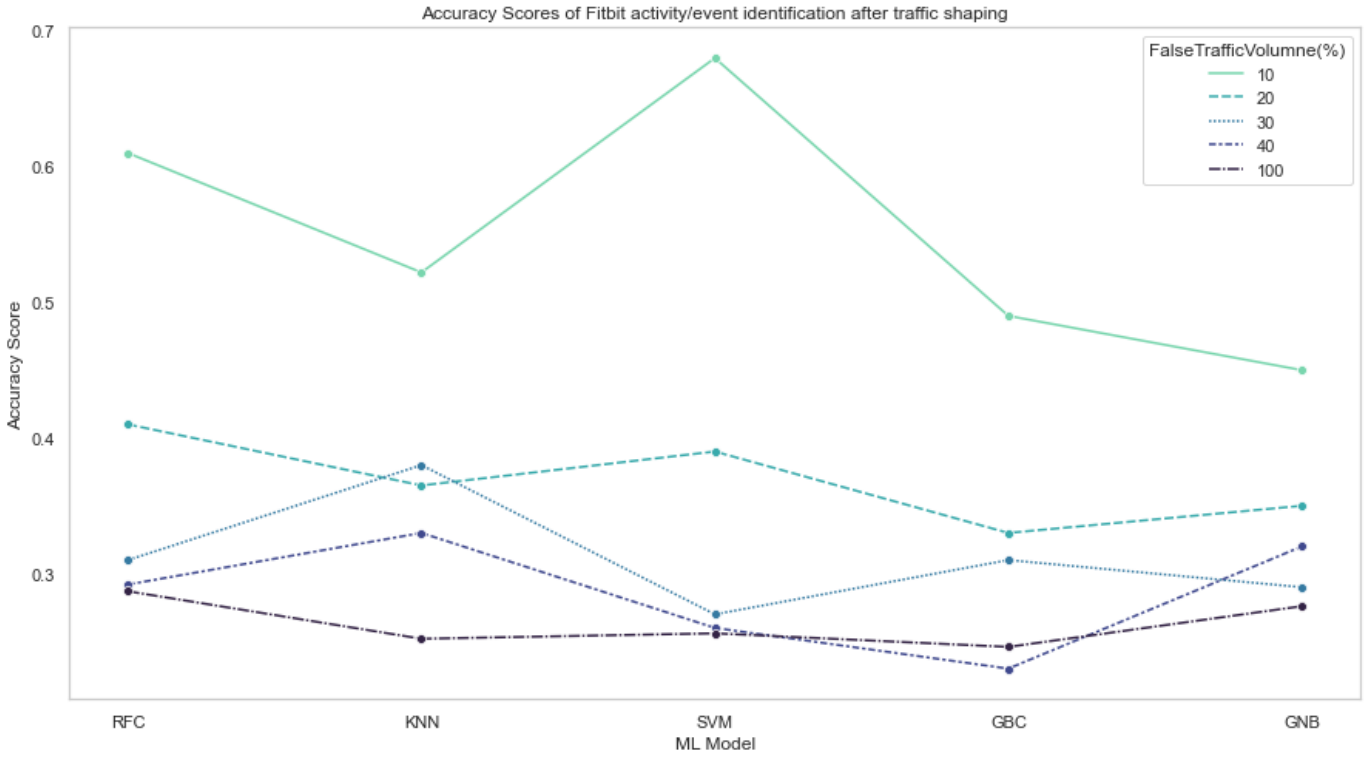

Figure 5.5: Accuracy of Fitbit activity/event identification after traffic shaping

We also plotted the average training time of event/activity identification classifiers to study the effect of false traffic injection on their training time. Figure 5.6 below shows the average training time taken by event/activity identification classifiers without and with 10% , 20%, 30%, 40% and 100% of false traffic injection to the original training data set. Training time for GBC, KNN and RFC increased by two times, by almost four times for GNB and by almost 2.5 times for SVM with addition of 40% false traffic.

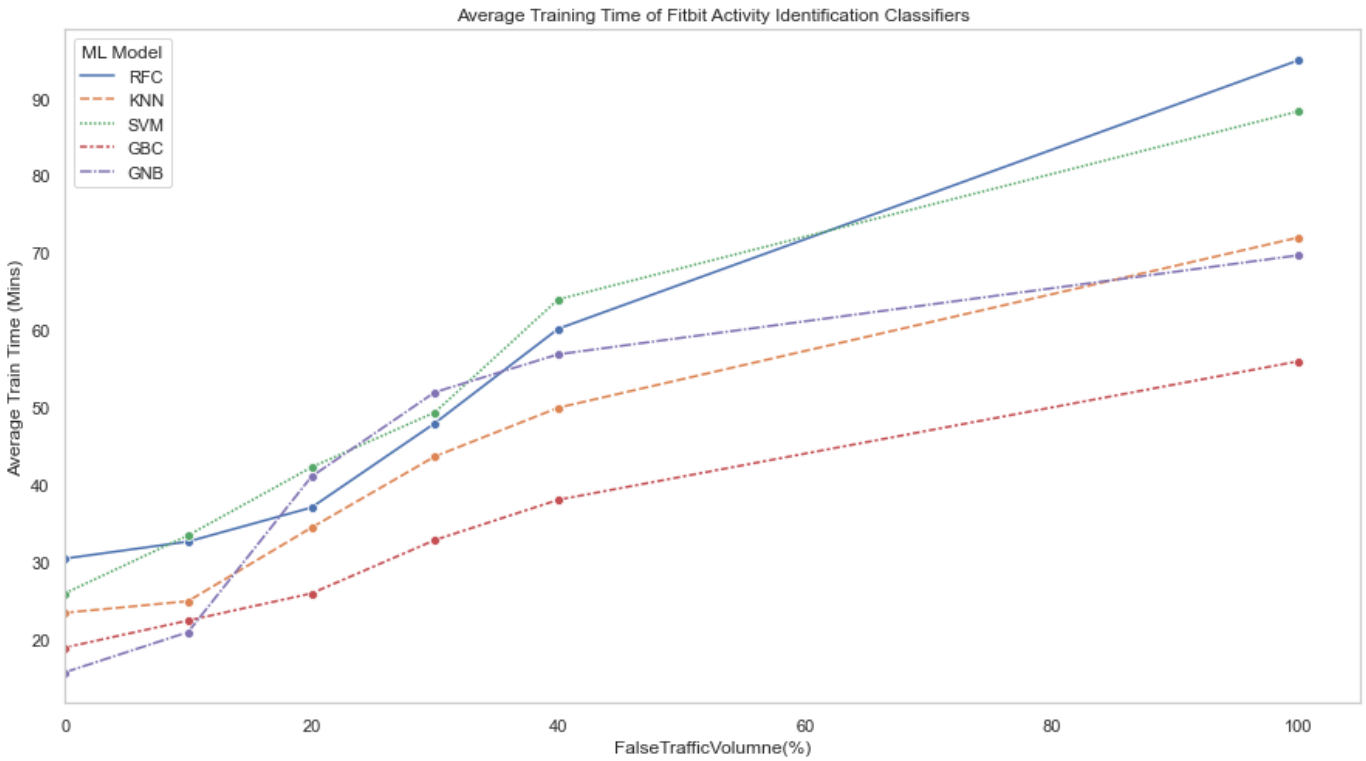

Figure 5.6: Average training time of Fitbit event/activity classifiers

5.4 Summary of Traffic Shaping Implementation

As seen in figures 5.3 and 5.5, we were able to further slightly reduce the accuracy of both device and activity/event identification classifiers by injecting exactly the same number of false traffic as the real traffic, however, doubling the training data set significantly increased (almost doubled) the training time of classifier compared to the training time with 30% and 40% false traffic volume as shown in figures 5.4 and 5.6. After observing the accuracy and training time of classifiers, we noticed that there was a trade-off between these two factors, and concluded that 30% and 40% of false traffic would be ideal for traffic shaping for device and activity identification respectively with acceptable increase in training time of classifiers. This proves that traffic shaping can prevent the risk of users' privacy violation using the traffic metadata attack.

CHAPTER 6
CONCLUSION AND FUTURE WORK

6.1 Conclusion

We were surprised by how easy it would be for a passive network observer to identify the IoMT devices being used by end users and infer user behavior even from encrypted traffic. ISPs already collect enough traffic rate information to perform the analysis we described. Because IoMT devices encode the physical world in network traffic, this presents a novel privacy threat to consumers. Regulatory agencies should keep this new context in mind when making rules governing ISP data collection and usage. The user interactions we analyzed in our case studies are directly related to the limited purpose of the device. For example, we concluded that traffic from a smart blood pressure monitor correlates to when a user measures blood pressure. Further research could allow an adversary to infer higher order behaviors, such as whether the user has a high or low blood pressure using machine learning techniques. However, this would require larger curated data sets with controlled experiments representing a wider range of user behaviors. More complex behaviors could also be inferred through the combination of traffic from multiple devices. We would like to reiterate that all of the analyses we performed required only send/receive rates of encrypted traffic to successfully identify user behavior. No deep packet inspection is necessary. A systematic solution for preserving consumer privacy would therefore require obfuscating or shaping all network traffic to mask variations that encode real world behavior. Ideally a solution would not negatively impact IoMT device performance, should respect data limits, and would not require modification of proprietary device software.

IoMT devices are becoming increasingly pervasive; however, the privacy concerns of owning many Internet connected devices remain insufficiently addressed. We analyzed four most popular IoMT devices and found that network traffic metadata and rates of all devices revealed the actual

devices users are using in their daily life and their own activities as well, making it apparent that encryption alone does not provide adequate privacy protection for smart medical devices. Given the generality of our traffic analysis strategy and the limited-purpose nature of most IoMT devices, we would not be surprised if many other currently available IoMT devices suffer similar privacy vulnerabilities. Also, given the effectiveness of traffic rate privacy attacks on all tested devices, especially with the use of machine learning techniques, we believe that IoMT device users should be concerned about traffic rate metadata attacks across all types of IoMT devices. And we hope that end users/consumers will become better aware of these privacy vulnerabilities.

We also design privacy defense techniques and show that traffic shaping can significantly reduce the private information like device type, state and user activities leaked through IoMT device network traffic data. Our traffic algorithm uses false traffic injection at regular time intervals to limit the information revealed about device and user activities through traffic rate metadata. It also conveys false information about device usage to the network observer. For instance, the injected fabricated traffic indicates the device is currently in use, which is not true at all. This is our main goal to evade passive network observers and attackers from correctly identifying device state or user activities. We demonstrated our traffic shaping implementation using IoMT network traffic traces of 4 commercially available IoMT devices.

6.2 Future Work

While we have demonstrated potential leakage of users' private information by applying machine learning classification algorithms on traffic and its metadata, and simulated the privacy protection using traffic shaping algorithm, there are other important aspects that still need to be addressed. Therefore, in our future work we plan to train and test classification models for several more types of IoMT devices. We will also apply model fine-tuning techniques to improve the classifiers performance. We will apply distribution like Laplace, Gaussian while injecting falsified traffic and implement our traffic shaping algorithm on real world setup.

BIBLIOGRAPHY

[1] Accuracy score metrics. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html. [Online; accessed 19-July-2022].

[2] F1 score metrics. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html. [Online; accessed 19-July-2022].

[3] Fitbit smart watch. https://www.fitbit.com/global/us/products/smartwatches. [Online; accessed 19-July-2022].

[4] Gaussian naive bayes classifier. https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html. [Online; accessed 19-July-2022].

[5] Gradient boosting classifier. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html. [Online; accessed 19-July-2022].

[6] K-nearest neighbor classifier. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html. [Online; accessed 19-July-2022].

[7] Kardia ekg monitor. https://store.kardia.com/products/kardiamobile6l. [Online; accessed 19-July-2022].

[8] Qardioarm smart blood pressure monitor. https://www.qardio.com/qardioarm-blood-pressure-monitor-iphone-android/. [Online; accessed 19-July-2022].

[9] Random forest classifier. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. [Online; accessed 19-July-2022].

[10] Support vector machine. https://scikit-learn.org/stable/modules/svm.html. [Online; accessed 19-July-2022].

[11] Visualbeat heart rate monitor. https://getwellue.com/pages/visualbeat-heart-rate-monitor. [Online; accessed 19-July-2022].

[12] Abbas Acar, Hossein Fereidooni, Tigist Abera, Amit Kumar Sikder, Markus Miettinen, Hidayet Aksu, Mauro Conti, Ahmad-Reza Sadeghi, and Selcuk Uluagac. Peek-a-boo: I see your smart home activities, even encrypted! In *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pages 207–218, 2020.

[13] Mohammed Ali Al-Garadi, Amr Mohamed, Abdulla Khalid Al-Ali, Xiaojiang Du, Ihsan Ali, and Mohsen Guizani. A survey of machine and deep learning methods for internet of things (iot) security. *IEEE Communications Surveys & Tutorials*, 22(3):1646–1685, 2020.

[14] Mohammad Amiri-Zarandi, Rozita A Dara, and Evan Fraser. A survey of machine learning-based solutions to protect privacy in the internet of things. *Computers & Security*, 96:101921, 2020.

[15] Noah Apthorpe, Danny Yuxing Huang, Dillon Reisman, Arvind Narayanan, and Nick Feamster. Keeping the smart home private with smart (er) iot traffic shaping. *arXiv preprint arXiv:1812.00955*, 2018.

[16] Noah Apthorpe, Dillon Reisman, and Nick Feamster. A smart home is no castle: Privacy vulnerabilities of encrypted iot traffic. *arXiv preprint arXiv:1705.06805*, 2017.

[17] Noah Apthorpe, Dillon Reisman, Srikanth Sundaresan, Arvind Narayanan, and Nick Feamster. Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic. *arXiv preprint arXiv:1708.05044*, 2017.

[18] Simon Birnbach and Simon Eberz. Peeves: Physical event verification in smart homes. 2019.

[19] Anat Bremler-Barr, Haim Levy, and Zohar Yakhini. Iot or not: Identifying iot devices in a short time scale. In *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9. IEEE, 2020.

[20] Anil Chacko and Thaier Hayajneh. Security and privacy issues with iot in healthcare. *EAI Endorsed Transactions on Pervasive Health and Technology*, 4(14):e2–e2, 2018.

[21] Rohan Doshi, Noah Apthorpe, and Nick Feamster. Machine learning ddos detection for consumer internet of things devices. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 29–35. IEEE, 2018.

[22] Kevin P Dyer, Scott E Coull, Thomas Ristenpart, and Thomas Shrimpton. Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail. In *2012 IEEE symposium on security and privacy*, pages 332–346. IEEE, 2012.

[23] Roman Kolcun, Diana Andreea Popescu, Vadim Safronov, Poonam Yadav, Anna Maria Mandalari, Richard Mortier, and Hamed Haddadi. Revisiting iot device identification. *arXiv preprint arXiv:2107.07818*, 2021.

[24] Raymond ST Lee. *Artificial intelligence in daily life*. Springer, 2020.

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

[26] Yongxin Liu, Jian Wang, Jianqiang Li, Shuteng Niu, and Houbing 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.

[27] Yair Meidan, Michael Bohadana, Asaf Shabtai, Juan David Guarnizo, Martín Ochoa, Nils Ole Tippenhauer, and Yuval Elovici. Profiliot: a machine learning approach for iot device identification based on network traffic analysis. In *Proceedings of the symposium on applied computing*, pages 506–509, 2017.

[28] Yair Meidan, Michael Bohadana, Asaf Shabtai, Martin Ochoa, Nils Ole Tippenhauer, Juan Davis Guarnizo, and Yuval Elovici. Detection of unauthorized iot devices using machine learning techniques. *arXiv preprint arXiv:1709.04647*, 2017.

[29] Markus Miettinen, Samuel Marchal, Ibbad Hafeez, N Asokan, Ahmad-Reza Sadeghi, and Sasu Tarkoma. Iot sentinel: Automated device-type identification for security enforcement in iot. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 2177–2184. IEEE, 2017.

[30] Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Hossein Fereidooni, N Asokan, and Ahmad-Reza Sadeghi. Dïot: A federated self-learning anomaly detection system for iot. In *2019 IEEE 39th International conference on distributed computing systems (ICDCS)*, pages 756–767. IEEE, 2019.

[31] Sode Pallavi and V Anantha Narayanan. An overview of practical attacks on ble based iot devices and their security. In *2019 5th international conference on advanced computing & communication systems (ICACCS)*, pages 694–698. IEEE, 2019.

[32] Jingjing Ren, Daniel J Dubois, David Choffnes, Anna Maria Mandalari, Roman Kolcun, and Hamed Haddadi. Information exposure from consumer iot devices: A multidimensional, network-informed measurement approach. In *Proceedings of the Internet Measurement Conference*, pages 267–279, 2019.

[33] Pieter Robyns, Eduard Marin, Wim Lamotte, Peter Quax, Dave Singelée, and Bart Preneel. Physical-layer fingerprinting of lora devices using supervised and zero-shot learning. In

Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks, pages 58–63, 2017.

[34] Arunan Sivanathan, Hassan Habibi Gharakheili, Franco Loi, Adam Radford, Chamith Wijenayake, Arun Vishwanath, and Vijay Sivaraman. Classifying iot devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, 18(8):1745–1759, 2018.

[35] Syeda Manjia Tahsien, Hadis Karimipour, and Petros Spachos. Machine learning based solutions for security of internet of things (iot): A survey. *Journal of Network and Computer Applications*, 161:102630, 2020.

[36] Mathy Vanhoef and Eyal Ronen. Dragonblood: Analyzing the dragonfly handshake of wpa3 and eap-pwd. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 517–533. IEEE, 2020.

[37] Liang Xiao, Yan Li, Guoan Han, Guolong Liu, and Weihua Zhuang. Phy-layer spoofing detection with reinforcement learning in wireless networks. *IEEE Transactions on Vehicular Technology*, 65(12):10037–10047, 2016.

[38] Liang Xiao, Xiaoyue Wan, Xiaozhen Lu, Yanyong Zhang, and Di 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.

[39] Keyang Yu, Qi Li, Dong Chen, Mohammad Rahman, and Shiqiang Wang. Privacyguard: Enhancing smart home user privacy. In *Proceedings of the 20th International Conference on Information Processing in Sensor Networks (co-located with CPS-IoT Week 2021)*, pages 62–76, 2021.

[40] Jaeseok Yun, Il-Yeup Ahn, JaeSeung Song, and Jaeho Kim. Implementation of sensing and actuation capabilities for iot devices using onem2m platforms. *Sensors*, 19(20):4567, 2019.