TU WIEN Informatics

Smart Home Privacy Leaks

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering & Internet Computing

eingereicht von

Simon Treiber, BSc.
Matrikelnummer 01326769

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Privatdoz. Mag.rer.soc.oec. Dipl.-Ing. Dr.techn. Edgar Weippl
Mitwirkung: Univ.Lektor Dipl.-Ing. Dr.techn. Georg Merzdovnik, BSc.

Wien, 20. Mai 2021

Simon Treiber	Edgar Weippl

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

Smart Home Privacy Leaks

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering & Internet Computing

by

Simon Treiber, BSc.
Registration Number 01326769

to the Faculty of Informatics

at the TU Wien

Advisor: Privatdoz. Mag.rer.soc.oec. Dipl.-Ing. Dr.techn. Edgar Weippl
Assistance: Univ.Lektor Dipl.-Ing. Dr.techn. Georg Merzdovnik, BSc.

Vienna, 20th May, 2021

Simon Treiber	Edgar Weippl

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Simon Treiber, BSc.

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 20. Mai 2021

Simon Treiber

Kurzfassung

Obwohl sich Smart-Home Technologien immer größerer Beliebtheit erfreuen dürfen und in immer mehr Haushalten zu finden sind, zeigen Forschungen, dass sich User:innen üblicherweise der Auswirkungen, die die gesammelten Daten solcher Geräte haben können, nicht bewusst sind. Weiters wissen User:innen oft nicht welche Daten von Smart-Home Geräten aufgezeichnet werden und an welche Unternehmen diese dann übertragen werden. Da diese übertragenen Daten von Smart-Home Geräten üblicherweise verschlüsselt sind, ist es ohne weitere Maßnahmen, wie Man-in-the-Middle-Angriffen nicht möglich den Inhalt der gesendeten Pakete zu lesen. Auch wenn die Daten unverschlüsselt gesendet werden, ist es auch keine einfache Aufgabe automatisiert zu ermitteln welche Daten gesendet werden. Dies liegt an der schieren Menge der unterschiedlichen Smart-Home Geräte unterschiedlichster Hersteller, die keinen einheitlichen Standard für die Nachrichten der Smart-Home Geräte verwenden.

Um das Bewusstsein der Endnenutzer:innen diesbezüglich zu stärken entwickeln wir eine Erweiterung für das Pi-hole, ein netzwerkweiter Werbefilter, der üblicherweise auf einem Raspberry Pi installiert wird. Diese Software zeigt den User:innen an welche Informationen über sie oder ihr Verhalten durch die gesammelten Daten der Smart-Home Geräte abgeleitet werden könnten. Außerdem schafft es mehr Transparenz bezüglich dem Verhalten der Geräte indem die Software den auftretenden Netzwerkverkehr dieser Geräte anzeigt und analysiert zu welchen Firmen Informationen gesendet werden.

Da es wie vorhin erwähnt nicht einfach möglich ist, aus den gesendeten Daten abzuleiten welche Informationen von diesen Geräten gesendet werden, entwickeln wir ein Modell, das es erlaubt, basierend auf den Funktionen bzw. der eingebauten Sensorik eines Gerätes abzuschätzen, welche Daten dieses sendet und wie diese Informationen die Privatsphäre beeinflussen. Zusätzlich ist es damit möglich zu berechnen welchen Einfluss ein potenzieller Datenaustausch zwischen mehreren Firmen hat.

Die automatische Klassifizierung der Smart-Home Geräte übernimmt ein Machine-Learning Modell. Dafür sind von uns Methoden zu Datenaufbereitung passend für die Daten die auf dem Pi-hole gesammelt werden können, entwickelt worden und mit diesen aufbereiteten Daten sind drei verschiedene, zu diesem Zwecke üblicherweise eingesetzten Machine-Learning Modelle überwacht, trainiert und verglichen worden.

Abstract

Despite the ever-growing impact of smart home internet of things devices research has shown that users are usually not aware of privacy leaks, which occur in such an environment. Furthermore, users are typically not informed, which data types are collected from such devices and with which companies this data is shared. Since the majority of traffic originating from smart home devices is encrypted, it is not viable without, e.g. Man-in-the-middle attack to read the content of the transmissions. Even if the traffic is unencrypted, it is not a trivial task to automatically gain information which data is sent, since there are many vastly different devices from several different vendors, using multiple diverse standards to format their messages.

In order to improve privacy awareness, we developed a novel privacy empowering tool. It is based on the network-wide ad blocker Pi-hole and is typically deployed on a raspberry pi. This probe is able to infer potential privacy leaks i.e., which information could be derived about the residents or about their behaviour, from smart home devices in a home network. Furthermore, it provides transparency by displaying and analysing the occurring network traffic. Moreover, the companies involved in the data exchange with the smart home devices are shown.

To overcome the previously mentioned problem of gaining information data transmitted by smart home devices, a privacy leak model is developed. This model derives information, which is sent by such devices and further potential privacy leaks based on their functional device types. Additionally, this model allows to provide methods to calculate the impact of sharing data between companies.

The classification of smart home devices in the home network, is achieved by a machine learning model developed by us. For this reason, we created a feature extraction method suitable for the data provided by the Pi-hole and compared three machine learning methods, which are commonly used for this specific task.

Contents

CHAPTER 1

Introduction

1.1 Problem definition

In 2025, it is estimated that there will be approximately 75 billion Internet-of-Things devices [1]. The economic impact of Smart Home devices is estimated to reach between 200 and 300 billion US dollars by 2025 [2].

In the meantime, users are not aware of privacy threats regarding Smart Homes, although privacy leaks occur regularly [3]. To raise awareness, technical probes in home networks are used to measure network traffic produced by such devices. These measurements are usually shown in real time. Furthermore, the visualisations group the information according to the company receiving the data.
Due to the end-to-end encryption of most Smart Home devices, the content of its network traffic cannot be captured and read, therefore the data captured only consists of metadata. Hence, to analyse which data is most likely sent by a specific IoT device, its metadata must be augmented with further information. Besides this, pattern analysis of the metadata can reveal information about the data that is sent. To reliable derive further information about the content, the smart home devices must be identified or classified, respectively (e.g., light bulb, vacuum cleaner).

First of all, this work must provide a reliable device classification method. Besides this, to raise more awareness, additional analytical capability will be provided, like providing insights, which data is shared with whom and how often. The size of the privacy leak is also estimated. But most importantly, the privacy implications of data sharing between different companies, is analysed. Also, additional information, which could be derived by gaining access to further data, received from another manufacturer, is shown.

1.2 Expected results

The expected result of this work includes a model to estimate privacy consequences of Smart Home data, including the ramifications of sharing data from different device classes and between different companies. Secondly, a technique to reliably identify Smart Home devices should be developed. Finally, based on this model a technical probe is developed and tested.

The model to estimate the privacy leaks includes, besides the consequences of sharing data, a worst-case analysis, which leaks could occur if two companies would collaborate.

This technical probe is based on Pi-hole, a contemporary network-wide ad-blocker. This tool currently only shows how many DNS requests to ad providers are blocked and is enhanced by privacy-empowering features, which are mentioned above.

1.3 Methodological approach

To reach the expected results, the methodological approach consists of the following steps:

Firstly, existing technical probes, tools and literature are reviewed. Current methods to identify Smart Home devices and techniques to augment network traces (add sense; what data might be sent) are analysed.

Secondly, the necessary features of network traffic to classify Smart Home devices must be identified. Especially which features are available, without further tampering with the device, such as UPnP or mDNS requests. A model to infer what data might be sent by a specific Smart Home device is developed, and how this data will affect privacy leaks. Further the implications of combining data from multiple devices from multiple companies, sharing data with each other, is also taken in account.

Next is the development of a prototype, which is designed to raise awareness of privacy leaks in Smart Homes, therefore this tool is aimed for a broad audience. It builds on an existing tool, namely Pi-hole, using its capabilities to intercept DNS requests of such devices, as well as its simple set up process.

Finally, the model and prototype is evaluated by using labelled network data from Smart Homes.

CHAPTER 2

State-of-the-art

This chapter describes the current state-of-the-art and basics of privacy empowering technical probes for monitoring smart home network flows. Their approaches to visualizes and intercept the network traffic generated by smart home internet of things devices are described. Additionally, we recall the history of privacy taxonomy. Further state-of-the-art machine learning methods to classify smart home devices are described as well as the privacy implications occurring in such environments.

2.1 Privacy empowering tools

The installation and maintenance of technologies in their home is normally coordinated by the residents of the home. Some research probes deployed by researchers breaches this assumption. To comply with this assumption Tolmie et al. propose that deploying privacy empowering probes in private environments should be achievable by users [4]. Therefore, many research probes use plug-and-play solutions.

Such tools typically collect data flows created by devices in private networks and provide information of the network flows, like how much data is sent, or when data is sent. There are several ways to intercept the network flow e.g., by creating a Wi-Fi hotspot, ARP spoofing or providing a DNS server.

2.1.1 Design Goals

To empirically test design goals for such technical probes Seymor et al. developed a technical probe called Aretha [5].
The three key design goals for privacy-empowering technical probes are: Legibility (DG1), Interpretability (DG2) and Actionable Choices (DG3). To achieve DG1, the technical probe must provide live and historical data of all information flows and their destinations.

Further, such a technical probe should help the user to understand which data is sent and which privacy risks are associated with it (DG2) and provide meaningful actions to control the information flows (DG3).

They provide Legibility (DG1) by connecting the technical probe to the home router to gain internet access and by providing a wireless network, where the devices which should be monitored must be connected. To reduce the amount of data collected, and since the payload of the traffic is usually encrypted, only the packed headers are stored. The user interface to show the collected information is based on IoT Refine [5, 6]. It allows to see the exposure per company, geographic destinations and jurisdiction [6].
Other technical probes like IotInspector [7] using ARP spoofing (See Section 3.2.1) or by providing a DNS server like the Pi-hole [8] to collect the network data.

To provide an overview of which companies own the domains a device is connected to, Huang et al. derive this information using the domain names and the webXRay domain owner list [7, 9]. Further, they use a list for tracker and ad content providers to add this information. Moreover, depending on the port, the service can be shown e.g., an endpoint with port 123 is most likely a NTP server.

In order to help users interpret the collected information Seymor et al. [5], provides basic information about network privacy and background of the involved companies. The provided curriculum was tested with three iterations of online surveys. One finding of them, regarding DG2 was that the users not only wanted to know who can access the data generated by the smart home devices, but also what data and why this data is sent, further a balance between the privacy risks of data collection of different devices by large platforms and single destinations should be found.

Actionable Choices (DG3) should give users the possibility to black unwanted traffic, either by company or device [5]. This should help users to exert control over their smart home devices.

2.1.2 Labelling network data

To help users understand which data is sent by smart home devices, it is useful to provide transparency by showing what kind of data is sent by smart home devices, as shown by OConnor et al. [10]. Further to label the collected network data, Huang et al. developed a technical probe, which collects the DNS requests and responses, remote IP addresses and ports, aggregated flow statistics over a five second window, broadcast information like SSDP, TLS Client Hello messages, the MAC OUI [1] and the hashed MAC [7]. The devices are then manually labelled by the users. The label for each device (vendor or category or both) is then uploaded to their central server. To help users with the labelling an auto-complete list is provided for this feature. To protect other users in the same

[1] First 3 octets of the MAC address, identifies the manufacturer of the network chipset

network not aware of this tool, IoTInspector scans in unencrypted HTTP requests for the user agent strings and uses the FingerBank API [11], which takes the MAC OUI and five domains queried by the device, to detect if a device is potentially a general-purpose device. If this is the case the data is not uploaded. Before publishing a data set, the labels are validated.

Comparable to the FingerBank API, Antrope et al. have shown that only DNS Queries and the MAC addresses can be used to provide a device-type fingerprint, that can be used as device identification [12]. With additional traffic rates they even can infer user activities .

A different approach to label network flow is to classify the behaviour of smart home devices using machine learning. HomeSnitch focuses on providing transparency of the behaviour of different smart home devices in a home network, therefore the identification and classification of devices and the sent traffic is a key feature [10]. They compared three different machine learning algorithms, resulting in the selection of the best performing, namely the random forest model (accuracy of 99.69% ±.06, precision 94.66% ±1.21, F1 score 93.93% ±1.40).

Further, it has been shown by Acar et al., that from such classified data a simple model of user activities, like a user entering his smart home and going to the bathroom, can be created [13]. For this analysis they incorporated a smart lock, smart lightning, motion and door sensors. Additionally, to Ethernet traffic Peek-a-Boo incorporates data from other commonly used technologies within IoT devices, like ZigBee and Bluetooth LE. Due to the encryption of the traffic, they use the MAC addresses to narrow down the device type. Moreover, the traffic rates of a device can further narrow down the device type.

2.1.3 Traffic blocking

Privacy empowering tools block specific traffic mainly for two different reasons. First to block ads respectively trackers and second to block unwanted traffic, as described by DG3 or even security threatening traffic.

For example, HomeSnitch provides a framework to define policies which data should be blocked, because they assume smart home devices lack good security protocols [10]. Further, they assume that their policies can prevent the exploiting of unpatched security vulnerabilities and logins with default credentials, or simple, their set of policies should prevent unwanted device behaviour. Such policies are simple triples of (<vendor>, <model>, <activity>). They achieve an overall accuracy of 99.40% of labelling behaviour using 13 different features from the collected traffic, despite 76.22% of the traffic from tier test data set is encrypted. The selected features describe the behaviour of the smart home application like bytes sent to the server.

In contrast, Aretha uses the concept of firewalls to let users block traffic [5]. This is

chosen because, this concept is well-known by the participants. The user friendliness is ensured by providing a simple user interface using drop-down lists. An ad blocker like Pi-hole can simplify this task for the users, by providing a block list maintained by experts.
Therefore, Pi-hole uses its built in DNS server to block DNS requests towards tracker and ad providing domains [8]. This blocking is based on large lists containing such domains, further domains can be added manually for all or a subset of all devices. Therefore, unlikely other ad blockers, it is blocking the ads and trackers for each device using it as DNS server. It has not to be installed onto these different devices.

In contrast to the other privacy empowering tools, its purpose is not to give an insight of the generated traffic of each device connected to it, since, it only provides statistics for the entire network, especially what percentage of traffic is blocked, and which domains are most frequently blocked. This kind of data is available in real-time, as well as historical data, but this information cannot be grouped by device. There is also no information to which company a domain belongs to, or in which jurisdiction the corresponding company is located.

2.2 Device classification

Accordingly, to [14], most of the machine learning classifications, classify either by protocol, application, action, category (regarding security or QoS requirements) or by devices. However, the device classification distinguishes between different devices, not between functional classes of devices. Salman et. al provides in this review an overview of typical machine learning approaches in this field [14]. Some of these different methods are described below.

In contrast a new approach, to classify also unseen devices, [15] incorporates time-dependencies of the network traffic into their classification approach. They used a long short-time memory (LSTM) convolutional neural network, to differentiate between four classes, namely Hubs, Electronics, Cameras and Switches & Triggers. They extracted the packet count of different protocols, the total number, the number of received and sent packages, further they provide statistics of the packet length like min, max, mean, sum, standard deviation, skewness, variance, and kurtosis to the machine learning method. With time windows of five minutes, a 50 % overlap and half the data as training set, they identified the classes for 15 devices with an accuracy of 74.8 %. Reducing the classes to two, the accuracy increased to 96.7%, due to the inability to identify reliable Hubs, and other reasons, including the fact that only two Hubs are contained in the data.

Further to classify known devices reliably a multi-stage classification incorporating DNS queries and interval, ports, used cipher suits, flow characteristics, sleep time, as well as NTP interval can be used [16]. Stage-0 uses a Naive Bayes Multinomial classifier for ports, DNS names and cipher suits. Naive Bayes Multinomial classifiers are a well-known

and well-performing technique for text classification, where a large number of unique words are classified [17]. In Stage-1 the resulting class of bag and the confidence for the bag, calculated by Stage-0 is combined with the other characteristics and finally classified by a Random Forrest classifier [16]. The combined accuracy is 99.88% and the relative root squared error (RRSE) is 5.06%. The accuracy of the Stage-0 classification of the Domain names is 79.48% (RRSE 57.56%). Especially devices from same manufacturers can be hardly distinguished, because they are very likely to communicate with the same domains. An additional reason is that the used data set in many instances contains no DNS queries. The most information is gained from the flow volume metric resp. port numbers, followed by the domain-names, which have relatively high computational expenses to extract from packets. In contrast, sleep time and NTP timing contributes less information. Further this classification can be performed in real-time, when using low- and medium-cost attributes, with a slight loss of accuracy.

A similar approach, with a more constrained data set, namely only DNS queries, partially behind a NAT uses document retrieval techniques to fingerprint devices [18]. Likely this method can also hardly differentiate between different devices from the same manufacturer because such devices often have nearly the same or an identical DNS behaviour, some different devices run even the same firmware with slight deviations. To reduce false positives and to reduce matching time, they filtered out the 100 most popular domains (accordingly to Alexa.com) from the training set. To generate the frequencies of certain DNS requests per client, a time window longer as the time-to-life of DNS resource records should be selected. The vast majority of smart home devices using a TTL for DNS records below one hour, therefore this is a reasonable choice. To calculate a detection threshold, as well as to estimate false positives a large data set from a university campus based in the US with labelled non-IoT devices is used. The training was performed using labelled data recorded in an IoT lab containing network traces from 53 different devices, over a time period of approximately two months. [13]

2.3 Privacy Taxonomy

The first attempts to define the relationship between technology and privacy date back to the invention of the portable camera [19]. Since then, privacy is difficult to define, because it effects many aspects of life as well as cultural facets and is therefore perceived individually. Nissenbaum defines privacy as contextual integrity [20]. Fundamentally, every part of daily life, happens in diffident spheres (contexts) and is guided by norms of information flow. Further every sphere is governed by a distinct set of norms. The set of norms can be on one the hand explicit and specific (cf. Church), on the other hand variable and implicit and even incomplete. Moreover, privacy is a very broad topic from Big Brother to overprotective parentsall the way to the protection of one's genetic information [21]. But there can be models to identify and prioritize privacy risks [21].

One such privacy risk model from ubiquitous computing by Hong et al. [21] is based

on the same idea as security threat models. They emphasise nonetheless the difference between security and privacy, namely security as mechanisms to control who may use or modify stored data, and privacy as the ability to decide who is allowed to access and use the data. Another difference they state is that the security mindset differs very often in what would be needed to design a privacy-sensitive application, because the adversary model does not work in privacy context, since there the users likely know the other party, in contrast to a typical adversary.
This model should help to transform the abstract principles of trust levels into a concrete issue. The first part of this model helps to find privacy risks by social and organizational, as well as technological questions, like what kind of data is shared or how personal information is collected. To classify the kind of information, which is shared, they propose among other models the seven types of identify knowledge by Marx [22] used to identify people, like their real name, address or unique identifiers like their social security number. Moreover, pseudonyms that cannot be easily traced, their behaviour or appearance, including their web browsing habits are part of the seven types. Finally, we can know the social categorization like gender, age, religion, sexual orientation, health information, as well as the relationships to other persons about a person.
The secondly, part of this model helps to prioritize the privacy risks found in part one [21]. Therefore, the likelihood L of an unwanted disclosure of personal information, the damage D of such an event and the costs C of adequate privacy protection should be estimated using the qualitative classes: high, medium and low. To manage the privacy risks ordered by likelihood and damage, the model provides further questions to develop potential solutions for each risk.

A further problem is that privacy is very definition-resistant and therefore not easy to measure, although many security researchers tried to quantify it [23, 24]. Barkhuus provides, since privacy is difficult to measure, some pitfalls when comparing different studies in the field of privacy in HCI on the example of sharing locations [24]. One of their findings is that even the term location is interpreted in a broad range, depending on the context, from exact location to a city or country, therefore they recommend to other researchers e.g., not to ask about sharing the location, since this is not a discreet data type. Further they propose to design studies which use real user data or even a real system, instead of asking users of to evaluate potential situations. Finally, they suggest viewing privacy in a broader perspective not only concerns of users but additionally ask why they do not want to share specific personal data.

Moreover, privacy is typically defined in a non-networked world. To reveal privacy for the networked world, Palen et al. pinpoints to the discrepancy between the ephemeral real world and a permanent technological interconnected world [25]. In the first genre: The disclosure boundary between Privacy and Publicity, they mention a discrepancy if one is only participating passively or when the boundaries are no longer in one's control, like other people posting photographs of you, or the information revealed by your Google search queries. Secondly in a networked world the boundaries between yourself and others

often blur because technology compromises the competence to fully understand how one's actions appear to others. Finally, that temporal boundaries often do not apply in the networked world, since information stored by technology is not, as in the real world, ephemeral but permanent. Therefore, they propose that one should analyse privacy concerns of new technologies in a socio-economic context, as well as the historical context of privacy regulations. Further one should keep in mind that privacy management is a balancing act, and that this management can only take place in the possibilities provided by technology, which in turn are based on choices made by people with certain outlook. To additionally help understanding privacy risks, respectively the risk of a potential privacy leak, an introduction of taxonomies for privacy is supportive. Finn et al. refined the four categories of Clarke [26], privacy of the person, privacy of personal data, privacy of personal behaviour and privacy of personal communication, due to technological improvements, adding the privacy of thoughts and feelings, privacy of location and space and privacy of association (including group privacy), likewise they added to the privacy of personal data, the privacy of image [19].

Eckhoff and Wagner merged the communication and association into social life, because communication with someone automatically results in association with that person, further they think communication is more a medium, which when exploited can reveal information about the other types and should therefore not be limited to one category [27]. Secondly, they merged privacy of the person and privacy of thoughts and feelings to state of body & mind, considering the body and the mind are not separated [27, 28]. The resulting five types of privacy, are: Location, State of Body & Mind, Behaviour & Action, Social life, and Media [27].

Location is not describing the exact location, but how long one is at a specific location. Therefore, it can be implied if the location is for example your home or workplace. Further, in a smart city environment, information about social life can be inferred.

State of Body & Mind This type of privacy includes biometrics, data about a person's health condition, including mental health, as well as emotions, opinions respectively thoughts.

Social Life containing contents of social interaction online and in person, among metadata of such interactions. Such information can also infer data about health (interactions with a hospital) or opinions (conversations about politics).

Behaviour & Action beside actions and hobbies, this type includes purchase patterns. This data can be used for targeted advisement, moreover this data often allows to draw conclusions about all other types.

Media is composed of images, video, and audio of a person. Further face recognition or else voice recognition can be used to draw inferences from the images, videos, or audio recordings to social life.

2.4 Privacy implications

To understand the privacy implications, which come along with smart home internet of things devices, one must first define which data is sent by which device. This information is not readable though, because the device ecosystems are usually closed [29]. Further the data sent in the majority of transmissions, is encrypted. With the sheer amount of different smart home devices available, using different protocols, it is not feasible to automatically assess the content of the transmitted data, even if the traffic is not encrypted. For encrypted traffic, many devices do not allow to change the firmware to conduct experiments, like man-in-the-middle attacks to decrypt it, therefore no ground truth of which data is sent can be accomplished, Ren et al. state.

Therefore, along with supporting the user to interpret the monitored smart home traffic, the categorization of companies receiving data in first, support and third party is useful to estimate privacy implications [29]. They determine the party of the connection endpoint as follows: if the organization of the domain, assessed by a WHOIS request, matches or is related to the manufacturer of the device it is considered as first party, if not they manually search if the organization provides a content delivering network or is specialized in cloud services it is categorized as support party, if none of the above applies, it is considered as third party.
Further the exposed data should also be categorized, therefore a simple model consisting of *Stored data*, *Sensor data* and *Activity data* is proposed, where *Stored data* are identifiers for the device, along with personal identifiable information e.g. activity logs. The *Sensor data* category contains of data collected by the sensors built in in the smart home devices (e.g., video) and the *Activity data* is the information how a user has used a certain functionality of the device, like turning the light on, using the mobile phone app.

Another concept to infer which data can be derived from different sensors is originating form monitoring elderly people and their mobility at home. From the personal data generated by sensors, like smart home devices, which are remotely accessible and send across the internet an attacker, as well as the companies receiving this data can, depending on the number and type of the sensors, infer activities [30]. These activities typically, contain the six basic activities of daily life, the daily routines like bathing/ showering, dressing, eating, functional mobility, personal hygiene and continence [30, 31].

In contrast of the privacy awareness of users, if they are asked specifically about which data they would like to share, most of them are likely to deny data flows, which are leaking the activities taking place at home, as well as data containing demographics (e.g., age, gender) [32]. Further users are not likely to take a discount in exchange for their privacy and not for a decrease in privacy. Users are also less comfortable with a device which handles sensitive data, if the users are not aware of a data flow, if they do not provide their consent, if the data is used beyond primary purpose, respectively if the data is not securely handled.

To make the data flow visible, even if the smart home application does not show it, first of all the traffic between the corresponding device and cloud must be intercepted (See Section 3.2).

This coincides with the design goals of Aretha (Section 2.1.1). To show the data flow corresponds with providing DG1 legibility and to help the user understand if such sensitive data can be derived DG2 can provide models to estimate this. Obviously DG3 can provide actions to deny such flows to specific companies.

Current techniques to obstruct information gains from the network traffic for an adversary, who has the same capabilities as an ISP, but not the manufacturer, are blocking or tunnel traffic respectively shape traffic e.g., send fake traffic or only packets with the same size [12]. Obviously, this does not block any information going to the companies to which the data is sent.

But companies operating in the EU should have, appropriate privacy preserving mechanisms implemented e.g., anonymization respectively pseudo-anonymization, to de-link data with personal identifiable information, according to GDPR Article 9. This provision can ease many privacy concerns within a smart home. One example are the privacy implications of collected sensor data of different smart home devices to a third-party [33]. Due to the de-linking, the arising privacy implications are mitigated. Proposed mitigation strategies for a non-trusted vendor, would be to use an anonymizer at the edge of the home network, or a trusted third-party anonymizer, which alters the data before transmitting it to the vendor. If the vendor is trustworthy, they can anonymize the data.

Furthermore, Article 8 of the Charter of Fundamental Rights of the European Union enshrines the "right to protection of personal data" [34]. It covers all possible information – and therewith data – related to a person, not just sensitive data [35]. The protection also encompasses the elicitation of data as well as its processing [35].

2.5 Privacy implications estimation

Simple models to compare privacy leaks consists of four Tiers, starting at Tier-0, where an adversary can detect if a home is occupied or not, and if the residents are sleeping [36]. At the next level (Tier-1), rooms can be mapped to sensors, further on at Tier-2 the type of room can be identified (e.g., living room, kitchen) [36]. Lastly at Tier-3 the sensors are classified, where additional information activities in specific rooms can be identified [36]. A drawback of this model is that only occupancy and activities are taken in account, hence data like health-related information which is also collected by some smart home devices is missing.

A more sophisticated model also takes the house (H), including all IoT devices (C)

along with mobile devices (M) and backends (B) into account [37]. Further the users (U), communication channels (L), which data is collected (D) are processed as well as rules describing how data transmitted between different entities (P) influences this model [37]. This is formally described in a tuple (H, N, U, L, D, P), where the nodes $N = \{C \cup M \cup B\}$ and H is defined as rooms or areas in the home [37]. All nodes in N are capable of at least one feature, mobile devices in N further have the feature of being mobile and the backends are grouped if they are in the cloud or at the edge [37].
Users are categorized in three groups: data subject (people in the smart home, who are identified through the data), data controller (provider; in charge of collecting data) and data users (people who use the data, can be the owner of the device, as well as employees of the provider or third parties) [37].
Links are represented in a graph, where each edge is a link between a node and a user, therefore all possible paths between information are transferred between subjects, sender and receivers [37].
The data is described by a tuple (d_i, d_s, d_p, d_t), where d_i (data item) is the attribute collected (e.g. temperature), d_s is the data subject [37]. In privacy related data the data subject might be the user and otherwise the system [37]. The purpose of collecting and processing the data is defined by d_p and d_t denotes the time the information is stored (also including infinite) [37].
The policies how data is transferred (P) is denoted as tuple (lg_i, dp_i, s, r, c), whereas $lg_i \in L$ is the link group identifier, dp_i are the data permissions a set of pairs consisting of data item (d_i) and operation e.g. (temperature, read) [37]. The sender is represented by s and the recipient by r, c consists of conjunctions which represent the context [37]. The context can be among others, time, location, and transmission frequency [37].
Further a contextual factor are the destination countries and therefore different jurisdictions of the traffic of a smart home device [29].
To identify privacy threats, they are categorized according to Ziegeldorf et al. into: Identification, localization and tracking, profiling, linkage, privacy-violation interaction and presentation, inventory attacks and lifecycle transactions [37, 38].

2.5.1 Privacy leaks visualisation

Similarly, to Smart Home Internet of Things privacy policies, online privacy policies are often complicated and difficult to understand for customers. This is partially due to the amount of time it takes to read the policies and the fact that specific information is hard to find [39].

An approach to assist customers to understand such policies is by introducing a standardized and simplified label comparable with the Nutrition Facts label found on food [39]. This label simplifies the policy by categorizing it by types of information and how this type of information is used, as well as with whom this information is shared [39]. Further it is shown if one can opt-in or must opt-out or if its mandatory to share this information [39]. Since many participants were not aware of the terms opt-in and opt-out these terms are described in the legend [39].

Customers did not only understand it more accurately than a privacy policy, they also were more confident in understanding this standardized privacy label, further they were faster when comparing privacy policies of different companies [40].

CHAPTER 3

Background

In this chapter we discuss the basic smart home taxonomy used in this work, as well as the introduction of a functional classification of smart home devices. Moreover, the fundamentals of traffic interception in home networks are explained. Then we describe the possible retrieving of private information from different device classes. Finally, the current knowledge of privacy awareness of users of smart home devices is depicted.

3.1 Smart Home Taxonomy

Commonly, smart home internet of things applications are divided into four coarse grained classes: entertainment, energy, security, and health care [41]. Smart home devices can be classified in sensors, actuators, gateways and smart objects [41]. The typical classifications of Smart Home Devices, which are usually too coarse-grained to infer which data they are sending and often solely based on hardware specification [42]. To overcome these drawbacks, a functional classification of devices, introducing a hierarchical taxonomy (See Figure 3.1) is proposed by Bugeja et al. [42]. This classification is backed by data gathered from two big IoT product collection platforms (iotlist.co and smarthomedb.com). The first of the eight main categories of this classification are Energy and Resource Management, containing Climate Control, Lightning Systems, metering devices, window/ door controller and plugs. The next classes are Entertainment, Health as well as Networking and Utilities, which consists of Gateways, Hubs, and wireless range extenders. The Human-Machine Interface is subdivided in Remote Controls and Voice Command Devices. The remaining classes are Household Appliances and Kitchen Aids, Security and Safety, and finally a generic class containing Sensors.

A similar categorization, which also focuses on functionality, comes up with a comparable, but even more finely organized categorization e.g., they break the Lightning class further down to outdoor, local, central, navigation and security lighting [43]. But the high-level

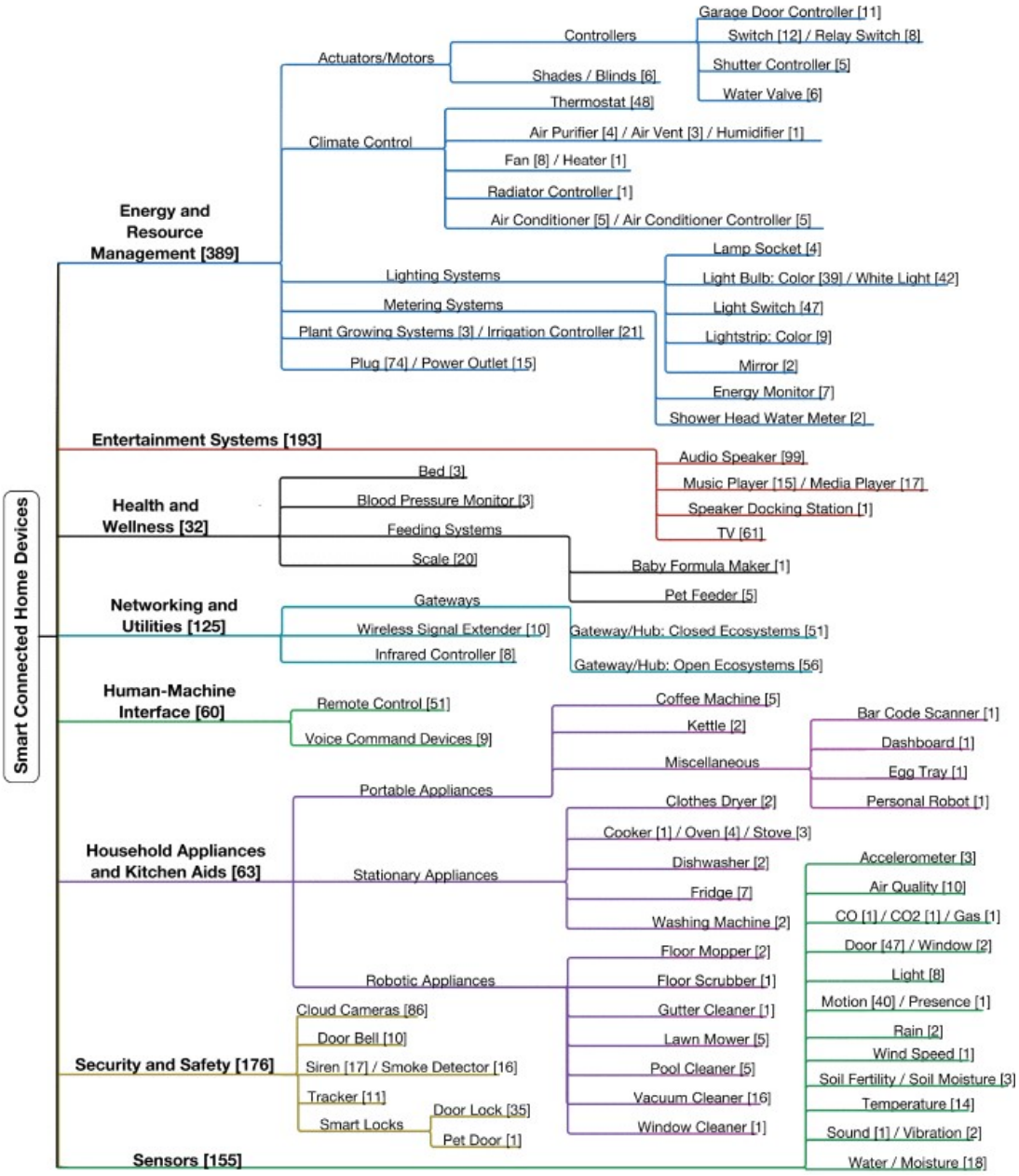

Figure 3.1: Functional device classes. The number inside the square brackets shows the number of devices in the data set. [42]

categories, are largely the same as the previous classification, although some classes are even further split, e.g., the Energy and Resource Management class is divided into Lightning, Shading system, Controlling of appliances, HVAC and Irrigation [42, 43]. Further this classification lacks a generic sensor class [43].

Smart home technologies can also be classified by their level of smartness. Ranging from level 1, simple analogue controlled devices with simple feedback, like a baby monitor or a television, to level 5 technology which automatically meets and anticipate all households needs [44]. Smart homes at the level 2 are using different devices in combination to provide household services, like heat (smart meter, thermometer, and heat pump) or entertainment (a TV controlled by a smart phone, playing content using its internet connection). Automation and anticipation of needs as well as the ability to program the smart home devices to meet specific necessities starts at level 3. A smart home at this level may turn the lights automatically off if nobody is home. Self-adaption and learning begin at level 4. An aggregation of smart homes over a neighbourhood, cities or even states can be labelled as level 6.

3.2 Traffic interception

To intercept the network communication in a home network, several techniques exist. In this subsection we describe three methods which are widely adopted by technical probes to intercept network traffic.

3.2.1 ARP-Spoofing

The address resolution protocol (ARP) translates protocol addresses (e.g., a IPv4 address) to the corresponding Ethernet address (MAC address) [45]. This protocol is used when sending packets to other hosts in the same . To achieve this translation a table is maintained with pairs of protocol addresses and Ethernet addresses [45]. If a device sends a packet to a host which does not appear in this table, it broadcasts an ARP-request for the Ethernet address for the protocol address in question [45]. The host with this specific protocol address, sends a reply directly to the device requesting it, containing its protocol and Ethernet address [45].

Many operating systems create a new entry in the ARP-table when an ARP-response is received and all create an entry after they sent a request to the host sending the response. Therefore, an attacker can listen for a specific ARP-request and send a reply with a non-corresponding Ethernet address and the requested protocol address or can send proactively an ARP reply with a desired Ethernet address, protocol address pair. Typically to intercept the traffic of an entire network an attacker sends ARP replies for the network gateway, with an Ethernet address of an attacker-controlled machine

within the network. This vulnerability is fixed in IPv6, by replacing the ARP protocol with the Neighbour Discovery Protocol (NDP) [46]. Therefore, if the Secure Neighbour Discovery (SEND) protocol security extension for NDP is used, this technique is not working in networks using IPv6, since the IPv6 addresses are cryptographically generated. This kind of addresses uses hashed public keys to generate part of the address. These announcements are then signed by the private key, to protect against spoofing.

Further many techniques to detect ARP-spoofing in IPv4 exist, firstly by altering the protocol or using a static ARP, secondly using intrusion detection systems or by using other tools from the TCP/IP stack like ICMP [47, 48, 49].

3.2.2 Wi-Fi router

A common technique around technical probes to collect and analyse network traffic is that they act as Wi-Fi router. This can be achieved by e.g., using a single-board computer with a Wi-Fi interface like the Raspberry Pi, or by using a router capable of installing custom software like OpenWrt [7].

In contrast to ARP-Spoofing this is not a Plug and Play solution, device which should be monitored have to be configured to use the network of the modified Wi-Fi router, but if a new device is installed in the smart home this configuration has to be done anyways.

3.2.3 DNS server

The network-based ad-blocker Pi-hole serves a DNS server for a home network and blocks DNS-requests for known ad providers[8]. This method allows to sniff all DNS requests of a client, which is configured to use Pi-hole as DNS server [8]. To use this DNS server, the router must be configured to use the Pi-hole for the DNS requests.

Due to privacy leaks from DNS to ISPs, some devices are using DNS over HTTPS, but Pi-hole can intercept this traffic by providing DNS over HTTPS capability [8, 50]. Further some devices use a fixed DNS server, which cannot be modified [51].

To classify smart home devices DNS queries can be used as a criterion [16]. The network behaviour, especially the DNS traffic generated by 14 different smart home devices is for example analysed by Y. Amar et. al [50]. They further used the DNS and DHCP logs to map devices, which randomize their MAC addresses like the iPad and resulting in fingerprinted devices [50].
Further broadcasted protocols can obviously be intercepted by such devices, most important packages using the simple service discovery protocol (SSDP).

DNS protocol

The DNS-Protocol is used to translate between domain names and IP addresses. Hence two different kind of DNS messages can be differentiated. First the DNS request to query the IP address of a specific domain. Secondly the DNS response, which contains additionally to the original request the answers to it. To differentiate these two messages the `QR` bit in the control field of the protocol is used; zero denotes the query.

The concrete DNS requests are stored in the question section, usually there is only one question per message, but the protocol allows more than one. The question is, exactly as the answers, stored as Resource Records (RR) but incomplete ones. In the DNS request RR the field's `name`, `class` and `type` are contained. The `name` contains the domain in question and the `class` is usually one since this encodes an Internet address. The common query `types` are:

- `A` - IP address
- `AAAA` - IPv6 address
- `CNAME` - canonical name; used for domain aliases
- `MX` - mail exchange

SSDP

The simple service discovery protocol is used by the universal Plug and Play (UPnP) protocol stack to determine which services or devices are available in the home network, without a central structure as used by the DNS system. It uses the HTTP over UDP to allow multi- and broadcast of such messages. This protocol uses the header of the HTTP, to discover and advertise devices.

The advertisement messages ueses the HTTP method `NOTIFY`, and is used to send alive, update and bye-bye messages, to inform other devices in the network that a service is available, changed or removed from the network. The alive and update messages contain a HTTP link in the `LOCATION` field, which allows to gather more information about the service.

To discover new services a discover request can be sent using the HTTP method `M-SEARCH` containing the desired service in the `ST` header. The `MX` header tells the clients to wait up to the stated seconds to respond. This should avoid an overload of the requesting node. The response to such a request contains similar information as the alive and update messages, like the `LOCATION` and additionally information about the operating system and product, respectively and its version in the `SERVER` header field.

3.3 Smart Home data collection

To estimate which data is collected by different smart home internet of things devices and which information can be derived therefrom, this subsection uses the state-of-the-art assumption, which information can be inferred from each functional device class.

Climate Control This device category consists of *Thermostats*, *Air Purifier/ Air Vent/ Humidifier*, *Fan/Heater* along with *Radiator controllers*. Devices of this category collect room climate data and are capable of detecting if the home is occupied, some even provide an *Auto Away* mode of operation [52, 53]. Further, the number of radiators in many countries equals the number of windows, therefore, the number of *Radiator controllers* is likely to match the number of windows.

Entertainment System Smart audio devices like, *Audio Speaker*, *Speaker Docking Station*, *Music Player/ Media Player* can gather personal information from the music respectively audio books and news which the device is playing [54].

Additionally, *smart TVs* collect viewing behaviour and commonly leak such information to the currently viewed TV-station by sending an ID via the HbbTV feature [55]. Some TVs actually incorporate a microphone and a camera, therefore such devices potentially leak even more personal data (compare: Camera, Voice Command Device) [55].

Smart Health and Wellness devices The different devices like *Blood Pressure Monitor*, *Scales* and *Beds* collect health related data, whereby *beds* are the most intrusive.

Smart beds collect health related data like the amount of sleep, the movement in bed, moreover it can analyse breathing patterns [56]. Most smart sleeping devices feature microphones, to monitor your sleep, especially they can warn one if snoring is detected, further heart rate sensors are standard [57].

Pet Feeder are the only device type in this category which does not collect health related data, but most of the commercially available feeders are equipped with a camera and often even with a microphone and speaker additionally [58, 59].

Camera With computer vision advancing, cameras are the closest to general-purpose sensing [60]. Further, a passive observer of the network traffic can differentiate if the camera feed is monitored or not, due to the tremendous size difference of the traffic [61, 62].

Household Appliances and Kitchen Aids Kitchen Aids like *Coffee Machines*, *Kettles*, *Dishwasher*, *Fridge* and *Cooker/ Oven/ Stove*. These devices all collect at least binary data (ON/OFF). A smart *fridge* can usually order food, and therefore imply how healthy your meals are [63].

Other household appliances include: *Cloths Dryer*, *Dishwasher* and *Washing Machines* which also collect binary data.
Further *Vacuum Cleaner/ Floor Mopper/ Floor Scrubber* which often map the home and therefore measure the square meters of each room.
Window Cleaner obviously can infer the number of windows in the home. Other devices in this category are: *Lawn Mower* and *Pool Cleaner*.

Door Bell Smart Door Bells often include cameras, to increase the safety awareness of users, as a downside this feature can be used to infer information about the user's social life, by identifying people who ring [64].

Energy monitor Energy monitors, like smart meters, can identify which household appliances are switched on, therefore they can derive the activities of daily live [65, 66]. Further it has been shown that smart meters with high resolution can identify audio-visual content, thus the TV watching behaviour of the user can be inferred [67].

Garage Door Controller Garage door controllers can alike smart locks leak occupancy and leaving and arrival times [66].

Hub Hubs typically allow, like Voice Command Devices, to control multiple devices, even from third-party vendors. The devices are typically assigned to their corresponding rooms. Hence a Hub with binary sensors, like lights or light switch respectively power outlets associated with their rooms, can not only infer occupancy but also activities of daily life [68].

Lighting System Lighting technologies as *Light Bulbs* and *Light Switches* transmit the ON/OFF state of the light, a potential privacy implication is that it can be inferred if a room is occupied [69, 70].

Plant Growing System Such systems are generally capable of measuring the temperature, therefore similar data as Climate Control devices acquire, are recorded (See Category Climate Control).

Plug/ Power Outlet Smart power outlets often measure electricity consumption, therefore leaking similar data as smart meters. Further they can act as ON/OFF switches and thus leak, equally as light switches, occupancy of a room.

Shades/ Blinds Obviously, the number of devices correspond with the number of windows, further if such devices are matched with rooms, with additional data from Climate devices activities can be inferred more precisely [52].

Water Meter Analogous to smart energy monitors, *Water valves* and *Shower Head Water Meters* can, depending on the sampling rate, detect occupancy, and even certain activities (like showering) as well as daily routines [71].

Smart Lock Smart locks can typically not only detect if a door is locked or not, but they can also further incorporate a door sensor and detect if the door is closed or not [72]. Therefore, such devices can infer if the home is occupied or not.

Smoke Detector Smoke detectors like the nest smoke alarm, are not only equipped with photoelectric and carbon monoxide sensors to detect smoke, in addition they have sensors to detect light and motion, therefore these smoke detectors are aware of lights switched on or off and if the room is occupied [73, 74].

Voice Command Device Information available to Voice Command Devices are manifold, for example the sleep patterns, by providing an alarm, as well as controlling the lights and house appliances [54]. If fitness tracker are linked to such devices, obviously health data too becomes available to these sort of devices [54]. Without such linkage health data can be collected through shopping lists and restaurant reservations [54]. Personal interests can be deduced from the music respectively audio book, which are played through the device, the search history and personal news and sport feeds [54]. Data about social life can be derived from the appointments stored in the calendar and your contact list used for calling [54].
Moreover, specific data is shared with linked third-party services, which users are often not aware of [75].

Air Quality It has been shown that CO_2 level sensors, as used in Air Quality sensors, are viable to detect occupancy [76]. Most air quality devices also containing microphones and light sensors, with this additional information occupancy detection is more reliable [76].

3.4 Information derivation

Since smart homes are typically a combination of different devices of diverse functional device classes, the ramifications of combining data from different classes are stated below.

Many Climate Control devices are capable to detect if the home is occupied, some even provide an *Auto Away* mode of operation, but if the size and layout of the room, as well as the sensor positions and windows are traceable, activities can also be inferred [52, 53]. The size and layout of a room can be e.g., measured by a smart vacuum cleaner, the number of windows can be likely matched with the number of radiator controllers. Using CO_2, humidity, temperature, and pressure levels, additionally an estimate, of how many people are in the room, can be made [77].

To monitor the activities of daily life, either cameras, microphones or an array of binary sensors (on/off switches) can be used [68]. Light Switches connected to a smart light bulb or a smart light switch, can act as binary sensor. It, without other sensors, can be used to detect presence [68]. If specified in which room the various light bulbs/switches are, the activity can be predicted [68].

With additional knowledge in which rooms the sensors, collecting environment data (Humidity, Pressure, Temperature) are, along with a light and position sensor, daily patterns of live can be inferred [78].

Microphones can be used to distinguish between different actives, especially if known in which room the microphone is installed (e.g., living room) [68].

3.5 Privacy awareness

Users usually are not very aware about potential implications and consequences of their privacy arising from smart homes, concludes a study conducted by Gerber et al., who recruited 1113 participants [3]. Most of the users cannot state a single mischievous consequence which results in sharing data with smart home respectively smart health devices. They commonly refer to general privacy issues like profiling, but not the consequences emerging from the data sharing. Most of the privacy consequences stated by the participants regarded online social networks. Further unknown privacy consequences might warrant educating users about privacy protection and leaks, as the aftermath of the Cambridge Analytical scandal and the "#deletefacebook"-campaign indicates.

Similarly, threat models created by users regarding smart homes are also very sparse [79]. Even if participants are aware of security or privacy issues, they often are not concerned about it since their threat models are often incomplete. The most common concern mentioned is physical security, therefore many participants used security cameras and were concerned about smart locks in their homes. Moreover, approximately half of the participants are not too concerned about privacy violations, which occur as a side-effect of using smart home devices, mainly they were anxious about audio or behaviour logs, as well as, but less often mentioned privacy risks which includes money or personal-identifiable information. On one hand, participants most commonly identify the companies of smart home devices as adversary, but on the other hand nearly all of them trust these companies and are not concerned about sharing data with them.
Reasons for the lack of concern are, as mentioned before: the trust in companies, participants think they are not important enough to be attacked, some think they have nothing to hide, or they believe to have secured their systems sufficiently, i.e., by using strong passwords [79]. Few participants see a trade-off between convenience, respectively functionality and security, respectively privacy risks.

In contrast most experts stated, in an interview conducted by Sovacool and Frurzyfer Del Rio, privacy, security and hacking as the number one smart home technology risk, followed by technical reliability, warranties and obsolescence [44]. Further they recommend a better customer protection regarding privacy and data security, including data control and restrictions in addition to encryption, clear guidelines of data ownership and safeguards against hackers.

Particularly in public spaces and even in smart homes (especially guests) people are often not aware which sensors are deployed. Therefore, to raise awareness Chow proposes a privacy stack [80]. The first part of this privacy stacked framework, the awareness part, deals with discovering all sensors and services connected to them, but because there is no standardization in doing so, this part is not trivial.

The next part deals with the inferences of the collected data, since a simple enumeration of collected data might not be sufficient for users to understand which information can be derived from such data. Therefore, Chow proposes to explicitly provide some basic inferences, but the users must understand that inferences might be more precise and even not complete, as more data is collected. Next users should be able to set preferences what data is shared with whom in which context. Because the framework is in the last stage and thus aware of the sensors in the close environment, it can notify users based on the preferences set up in the previous layer. This kind of notifications should support users to understand better the implications and interactions with nearby IoT services and their privacy respectively trust properties.

CHAPTER 4

Methodology

In this chapter we describe the research methods used in this master thesis. The first part provides an overview of the concepts and methods used to extract data from a network stream to classify devices. Further the concepts used to design a privacy leak model are introduced, as well as design goals for developing a technical probe used to display and analyse privacy leaks are mentioned.

Additionally, details of the data sets to validate and analyse the device classification, privacy leak model and the overall functionality of the technical probe are described. These two data sets consist of network traces from several smart home internet of things devices recorded over several days.

The last part contains analysis methods in order to compare different machine learning classification approaches with each other.

4.1 Concepts & Methods

The aim of this section is to outline the methods and concepts used to build the machine learning models and to prepare the network stream for these models. Next, we describe which method is used to design the privacy leak model. Finally, we introduce the design goals for our technical probe.

4.1.1 Device classification

The device classification is achieved by using machine learning techniques. Therefore, this part describes how the features are extracted and aggregated from the data. Furthermore, the different machine learning models used to classify smart home devices are specified. Salman et. al review several machine learning techniques to classify smart home devices

from network data, where different approaches are well performing [14]. Therefore, we compare three promising machine learning approaches and choose the best performing one for the technical probe.

Feature extraction and aggregation

Network data generated by smart home devices is usually collected using a network packet analyser like *tcpdump* or *scapy*. Due to the design of the technical probe, based on Pi-hole, the available network traffic features are limited. On the one hand broadcast packages and on the other hand DNS requests. Therefore, the feature extraction has to rely on packet-based characteristics, instead of flow-based characteristics.

Salman et. al mentions two suitable and widely adopt methods to extract features from the collected data, first to represent the data as time series and secondly as word embeddings [14].

Such time series can be constructed by using arriving packets and network flows or by using interarrival time of packets. Due to the lack of network flows, we concentrate on interarrival time of packets. Usually, several statistical features are computed from such time series and used as input for a classifier. These statistical features include length, median, mean, skewness, entropy, standard deviation, variance, as well as several wave transformations [14]. To obtain this statistical information, typically time windows with a fixed length T from such time series are analysed [15]. This method can be used e.g., to learn about different network time protocol behaviours, or to differentiate between similar DNS requests.

Word embeddings deal with word-based data sets, like DNS requests or HTTP User-Agent fields to classify devices [14]. Originating in natural language processing, Bag-of-Words represents words as vectors. The number of occurrences of a word in the bag is also taken in account. Usually special characters, as well as common words are filtered out before processing [16].

Machine learning methods

The classification problem is formally an optimization problem, where the minimization of a specific cost function is the aim to get the best class.
Due to the availability of labelled data sets, supervised machine learning methods can be applied. Supervised machine learning methods infer the rules or model parameters over the training phase. For this master thesis we use three supervised machine learning methods, namely Naïve Bayes, K-Nearest neighbour, and Random Forest.

Naïve Bayes

This probabilistic classifying method is based on Bayes' theorem, with a naïve independence assumption between features. Such models are very simple and fast to build, nonetheless such classifier perform well, when dealing with large bags of unique words [17].

Since the bag of words are multinomial features the multinomial naïve bayes classifier is using the following formula to calculate the probabilities:

$$P(w_j|c_i) = \frac{1 + \Sigma_{l=1}^{I} count(l, c_i, w_j)}{N + \Sigma_{l=1}^{N}\Sigma_{m=1}^{I} count(l, c_i, w_k)}$$

Where I is the total number of instances in the training set, N is the total of unique words, $count(l, c_i, w_j)$ counts for each instance l the occurrence of w_j if the class is c_i.

To calculate the class for an test instance, for each class the following probability is calculated and the most likely is chosen (maximum probability).

$$P(c_i|W) = P(c_i) \prod_{j=1}^{N} P(w_j|c_i)^{n(w_j,W)}$$

Where N is the total of unique words, W represents a test instance, containing the occurrences for every word, $n(w_j, W)$ is the number of occurrences of w_j in W.

K-Nearest neighbour (KNN)

The K-nearest neighbour algorithm is a lazy machine learning algorithm. Therefore, the only part of the training phase is to store the training set.
On classification of a test instance the distances between the K-nearest neighbours is calculated. The test instance is assigned to the majority of the classes of the k-nearest neighbours. A drawback of majority-voting is that frequent classes are likely to dominate the classification.
The weighted KNN classifier multiplies the neighbours by the distance to the test vector, to be more resistant against outliers. For discrete values, as in text classification, Hamming Distance is typically used to measure the distance between instances.
Important parameters for this classification are the number K of nearest neighbours and if the classifier should weight the neighbours.

Random Forest

This classifier is an ensemble machine learning method, creating multiple decision trees at the learning phase. Typically, only a subset of the features, as well as the instances of training data set is used to build each tree.
At the training phase, the classification is performed for each decision tree independent and averaged over all trees. The class with the maximum probability is selected by the Random Forest.

In a single decision tree, the knots represent a number of questions and the leaves a class. While answering the questions and navigating through the corresponding answers (edges), a test set is classified. This method though is vulnerable to overfitting [14]. Random forests overcome this drawback, by measuring the quality of the test data split, used to build a tree.

4.1.2 Privacy leak model

The privacy leak model is based on a privacy-centred system model for smart connected homes, which formally describes a smart home environment [37]. This model's purpose is to formally identify potential privacy threats, by applying the provided primitives to the model [37]. These primitives heavily rely on the policies of the model and categorize the potential threats by Ziegeldorf et. al enumeration of privacy threats [38].

This formal model is repurposed to infer privacy leaks, based on the devices (nodes) in this model. In contrast to the proposed model, the privacy leaks are not categorized by Ziegeldorf et. al enumeration of privacy threats, but rather by the five types of privacy [27].
Further this classification is solely based on the functional device ontology, which is also used as an overarching framework for the privacy-centred system model.
The generated data from each device class, is in contrast derived from the class and not specified by the data set. The data sent from each device class is based on typical sensors and features of devices within this class. This information cannot be inferred from the network traffic, due its encryption. Since the user interactions cannot be automatically assessed and are not needed to infer types of privacy, this set is omitted.

The implications inferred by the device type are based on basic research conducted in the smart home internet of things and smart elder care research.

4.1.3 Technical Probe

The design of the technical probe is based on the design goals proposed by Seymor et al. [5]. These three design goals are Legibility (DG1), Interpretability (DG2) and Actionable Choices (DG3).

- **Legibility (DG1)** overcome the non-transparency of connection end smart home IoT devices, the connection endpoints of these devices should be shown in real-time and historical records of the data should be provided [5].

- **Interpretability (DG2)**: Help users to understand privacy risks from the collected data, they propose on the on hand educational material to teach the basics of networked privacy risks and examples how companies handle their data [5]. But on the other hand they also conclude that the content of the data exchange, as well as privacy labels, improve the interpretability. Moreover, they propose the risk of big platforms accumulating data from several different devices should be emphasised.

- **Actionable Choices (DG3)** allows the user to block traffic between devices and companies [5]. This goal should be achievable by a simple user interface or by providing block lists. They further propose to use a third-party tool such a Pi-hole to fulfil this goal.

4.2 Data sets

To analyse the accuracy of the machine learning algorithms, as well as to test the privacy leaks model and the technical probe, we use two labelled IoT data sets.

The first data set was collected by IoTFinder on March 20, 21, 28 and from April 10^{th} to April 19^{th}, resulting in 13 days of collected network traffic from 65 devices, where 55 devices are smart home IoT devices [81, 82]. These devices can be classified in 14 different categories.

The second data set containing network traces from smart home devices was collected over six months of which 20 days from 23^{th} of September 2016 to 12^{th} of October 2016 are publicly available [16].
This data set contains traces of 21 different smart home IoT devices, from 13 different categories, as well as ten other devices, including tablets and phones, as well as the gateway and PCs [16].

These data sets are collected in lab environments with the intent of identifying IoT devices, containing all traffic features generated by the devices and sent through the router [81, 16].

The labels of these two data sets are the vendor and the model, since we want to classify devices by their functional device classes, we first must map each model to its corresponding functional device class. Therefore, we looked the model up on `www.smarthomedb.com`. The mapping for the two data sets is illustrated in Table 4.1 and Table 4.2. The non-smart home IoT devices for Table 4.1 are: Secuifi Almond, nVidia Shield, Nintendo Switch, PlayStation4, My Cloud EX2 Ultra, Xbox OneX, Ubuntu Desktop, Android

Tables, iPhone and an iPad. The provided IP addresses are further converted into the corresponding MAC addresses.

Model	Functional device class	IP address
Google On Hub	Hub	192.168.0.2
Samsung Smart Things Hub	Hub	192.168.0.4
Philips HUE Hub	Hub	192.168.0.5
Insteon Hub	Hub	192.168.0.6
Sonos	Voice Command Device	192.168.0.7
Nest Camera	Camera	192.168.0.10
Belkin WeMo Motion Sensor	Motion/ Presence	192.168.0.12
LIFX Virtual Bulb	Light Bulb	192.168.0.13
Belkin WeMo Switch	Plug/ Power Outlet	192.168.0.14
Amazon Echo Gen. 1	Voice Command Device	192.168.0.15
Wink Hub	Hub	192.168.0.16
Nest Protect	Smoke Detector/ Siren	192.168.0.17
Belkin Netcam	Camera	192.168.0.18
Ring Doorbell	Door Bell	192.168.0.19
Roku TV	TV	192.168.0.21
Roku 4	TV	192.168.0.22
Amazon Fire TV	TV	192.168.0.23
Apple TV Gen. 4	TV	192.168.0.25
Belkin WeMo Link	Light Bulb	192.168.0.26
Netgear Arlo Camera	Camera	192.168.0.27
D-Link DCS-5009 Camera	Camera	192.168.0.28
Logitech LogiCircle	Camera	192.168.0.29
Canary	Camera	192.168.0.30
PiperNV	Camera	192.168.0.31
Withings Home	Camera	192.168.0.32
Belkin WeMo Crockpot	Cooker/ Oven/ Stove	192.168.0.33
MiCasa Verde Vera Lite	Hub	192.168.0.34
Chinese Webcam	Camera	192.168.0.35
August Doorbell Cam	Door Bell	192.168.0.36
TP-Link WiFi Plug	Plug/ Power Outlet	192.168.0.37
Chamberlain myQ GarageOpener	Garage Door Controller	192.168.0.38
Logitech Harmony Hub	Hub	192.168.0.39
Caseta Wireless Hub	Hub	192.168.0.41
Google Home Mini	Voice Command Device	192.168.0.42
Google Home	Voice Command Device	192.168.0.43
Bose Sound Touch 10	Voice Command Device	192.168.0.44
Harmon Kardon Invoke	Voice Command Device	192.168.0.45
Apple Home Pod	Voice Command Device	192.168.0.47

Roomba	Vacuum Cleaner/ Floor Mopper/ Floor Scrubber	192.168.0.48
Samsung Smart TV	TV	192.168.0.49
Koogeek Lightbulb	Light Bulb	192.168.0.50
TP-Link Samrt WiFi LED Bulb	Light Bulb	192.168.0.51
Wink2 Hub	Hub	192.168.0.52
Nest Cam IQ	Camera	192.168.0.53
Nest Guard	Motion/ Presence	192.168.0.54
Nest Bell	Door Bell	192.168.0.55
Rachio 3	Plant Growing System	192.168.0.56
Amazon Echo Dot Gen. 3	Voice Command Device	192.168.0.57
Nest Thermostat	Thermostat	192.168.0.58
Google Home Hub	Voice Command Device	192.168.0.59
Facebook Portal	Camera	192.168.0.62
Sonos Beam	Voice Command Device	192.168.0.63
LG WebOS TV	TV	192.168.0.64
Axis Network Camera	Camera	192.168.0.65
AV Tech IPCam	Camera	192.168.0.67

Table 4.1: Mapping of device model and class for the data set provided by [81]. Excluding Non-IoT devices

The devices classified as Non-IoT excluded in Table 4.2 are a HP Printer, a Samsung Galaxy Tablet, two Android Phones, two Mac Books, a Laptop, as well as the Gateway.

Model	**Functional device class**	**MAC address**
Smart Things	Hub	d0:52:a8:00:67:5e
Amazon Echo	Voice Command Device	44:65:0d:56:cc:d3
Netatmo Welcome	Camera	70:ee:50:18:34:43
TP-Link Day Night Cloud Camera	Camera	f4:f2:6d:93:51:f1
Samsung SmartCam	Camera	00:16:6c:ab:6b:88
Dropcam	Camera	30:8c:fb:2f:e4:b2
Insteon Camera	Camera	00:62:6e:51:27:2e
Withings Samrt Baby Monitor	Camera	00:24:e4:11:18:a8
Belkin Wemo Switch	Plug/ Power Outlet	ec:1a:59:79:f4:89
TP-Link Smart plug	Plug/ Power Outlet	50:c7:bf:00:56:39
iHome Power Plug	Plug/ Power Outlet	74:c6:3b:29:d7:1d
Belkin WeMo Motion Senor	Motion/ Presence	ec:1a:59:83:28:11
Nest Smoke Alarm	Smoke Detector/ Siren	18:b4:30:25:be:e4
Netatmo weather station	Air Quality	70:ee:50:03:b8:ac
Withings Smart scale	Scale	00:24:e4:1b:6f:96

Blipcare Blood Pressure meter	Blood Pressure meter	74:6a:89:00:2e:25
Withings Aura smart sleep sensor	Sleep senor	00:24:e4:20:28:c6
Light Bulbs LiFX Smart Bulb	Light Bulb	d0:73:d5:01:83:08
Triby Speaker	Audio Speaker	18:b7:9e:02:20:44
PIX-STAR Photo-frame	Photo-Frame	e0:76:d0:33:bb:85
Nest Dropcam	Camera	30:8c:fb:b6:ea:45

Table 4.2: Mapping of device model and class for the data set provided by [16]. Excluding Non-IoT devices

4.3 Analysis

To compare the results of the machine learning models with each other and models proposed by different researchers, based on the same data sets, the following measurements will be taken: the model accuracy, the F1-score and the model precision and recall.

To interpret how reliably each class is classified, a confusion matrix is provided.
Further, each model will be cross validated.

- Model **accuracy** is the ratio between correct predictions and absolute predictions. In a multilabel classification, a predicted label must match exactly with a correct label, to be counted as correctly classified.
- Model **precision** is the division of the true positives by the sum of true positives and false positives.
- Model **recall** is a value between 0 and 1 calculated by the division of the true positives by the sum of true positives and true negatives.
- **F1-score** is for multiple classes the weighted average of all F1-scores of every class. The f1 score of one class is the weighted average between recall and precision: $F1 = 2 * \frac{p*r}{p+r}$, where p denotes the precision and r the recall of the model.

Cross-Validation

The usage of the same data to train and test a model is a methodological mistake because the model will be overfitted, i.e., it would just repeat the learned labels. The k-cross-validation technique therefore splits the data in k smaller sets, using k-1 sets to train the model and the remaining set to test it. The performance of the model is then measured by taking the average of all results. This technique helps to better understand the performance of a classifier, since the parameters are less likely to be biased by the training and test sets.

CHAPTER 5

Implementation

First, the software architecture of the technical probe is described. This technical probe uses as basic building block the Pi-hole [8] a network-wide ad blocker, running on a Raspberry Pi. It enhances its functionality by providing a classification of Smart-Home IoT devices, as well as an overview of the network activity of specific IoT devices. Further the network activities are analysed regarding destination country and company. Moreover, an estimation of the privacy implications caused by a device is provided. Finally, a cross company analysis of the privacy leaks resulting in having access to data collected by multiple devices is provided.

The classification functionality is provided by a machine learning model. The data extraction and aggregation to train, test and analyse the three different machine learning models, is implemented separately. The best performing model is provided to the corresponding component of the add-on of the Pi-hole.

5.1 Software Architecture

The augmentation of the Pi-hole [8], contains the following components:

- traffic capturing and augmentation service
- device classification
- privacy leaks estimation

Additionally, an UI is integrated to the Pi-hole's UI to visualize the privacy leaks.

The back end of this probe is written in Python 3.7 [83], the front-end is based on

the web-UI provided by Pi-hole. This UI is written in PHP and JavaScript and the data is stored by the Pi-hole core in an SQLite database. To stick with this convention, the captured traffic and further results of the additional features are also stored in SQLite databases as well.

The back end consisting of the traffic capturing and augmentation and device classification is provided as a Unix-service, which automatically starts on system start-up. Further to store unsuccessful identifications of organizations a Redis database is used.

In contrast, the analysis of the privacy implications and the impact of multiple companies sharing data with each other, is loosely coupled to the back end of the other parts. It consists of a back end, which is implemented in PHP. It is only calculated if a user selects multiple companies and starts this analysis via the web-UI.

Finally a python script to test the software with parts of the data sets is implemented.

5.2 Traffic capturing and augmentation

To further process the incoming traffic, additionally to the data collection provided by Pi-hole, the traffic is intercepted using the `scappy` package for python. In more detail the DNS requests sent to the Pi-Hole and broadcasted Simple Service Discovery Protocol (SSDP) packages are processed and stored.

The Pi-hole iteself sends DNS requests to the preconfigured DNS servers. Therefore, every DNS request is intercepted multiple times, but these are filtered and not further processed. Because scapy is known to miss some packages if the load is high, this filtering is done by providing this filter to scapy, to reduce the number of packages, which are intercepted, drastically. The filter further checks if the packet is a DNS or SSDP packet in detail *src host not <pihole ip> and (udp port 53 || udp port 1900)* is used, where *<pihole ip>* is replaced with the actual IP of the Pi-hole, which is automatically assessed. Because DNS answers also containing the DNS query information, a DNS packet must further be examined, if it contains DNS response data, then the packet is ignored, because such packages do not originate from the smart home devices. They are rather generated by Pi-hole.

Furthermore, to decrease the missed packages, due to the duration the augmentation takes, the augmentation part is processed in a new thread. To limit the number of concurrent threads, a thread pool with the maximum of ten concurrent threads is used.

Moreover, for each DNS request the organization is identified (See Section 5.2.1) and the status of the request is copied from the Pi-holes' database to determine if the domain in question is classified as ad or tracking provider by the Pi-hole.

SSDP packages are no further processed in this part, the content and timestamp of such packages are stored and assigned to the device which is the source of the package.

To ease the device identification for the user, on the first encounter, devices are named, using the manufacturer fixed by the Organizationally Unique Identifier of the MAC-address and their IP address. The manufacturer is provided by the Pi-hole. This name can be changed by the user. After the device is successfully classified, the device class is also provided.

5.2.1 Identify Organizations

To identify the organizations three different sources are used in the following order. If the organization cannot be identified, at least the location of the domain is assessed. To consistently augment domains from each DNS request, first of all the domain name is extracted.

webXray

At the first start-up of the service, the webXray list, containing the top approximately 1000 websites with their owners and the location of the owning organization, aggregated by T. Libert [84], is red and a database of its domains and owners is created. As this list contains the subsidiaries owning the domain, with a reference to the parent company or an intermediate company we pre-process this list, because we want to identify the parent company of each domain.

WHOIS

If the domain is not yet contained in this database a WHOIS request of the domain in question is parsed. For such requests not a single database for all domains exists, thus some entries only contain links to the WHOIS server storing the information of the domain in question. To determine the owner of the domain the `Registrant Organization` is used, for the location the `Registrant Country`. Since this is a wide-used technique, an existing python library (`python-whois`) is available to parse and send such requests. To avoid redirects to other whois servers, this library contacts the specific whois server responsible of the top-level domain in the domain.

Since the enforcement of the GDPR personal identifiable information is often hidden behind privacy proxies. This also contains information for some organizations and location of these organizations. One such response of a WHOIS request of a domain whose Registrant Organization is hidden behind a privacy proxy is shown in Figure 5.1. Therefore, if the request is successful, the results must be checked if the organization is

masked by a privacy proxy. This filtering is achieved by a list of known privacy proxies. Further organizations containing the words *retracted*, *privacy* or *proxy* are retracted.

```
Domain Name: NESSUS.ORG
Registry Domain ID: D1234225-LROR
Registrar WHOIS Server: whois.registrar.amazon.com
Registrant Organization: Whois Privacy Service
Registrant State/Province: WA
Registrant Country: US
```

Figure 5.1: Essential parts of a WHOIS lookup of a domain using a privacy proxy

bigpicture.io

If the WHOIS fails or the information is not available, the *bigpicture.io* web service API is queried to gain information about the company owning the domain. This service is used as last source, because the free service allows only a limited number of 1,000 requests per month.
Their domain enrichment API allows to assess geographical and address information, the name of the company, the logo of the company, as well as the sector of the company. Further metrics of the company like revenue, number of employees, social media accounts and their domain aliases are provided. The essential parts of a sample response querying `amazon.com` can be seen in Figure 5.2. In this figure the social media data, description and company metrics are omitted.

Geoip

Finally, if all sources mentioned above have no information about the domain, the location of the IP is assessed using the *geoip* python package and the domain is used as company name. The *geoip* package uses the predefined IP address spaces to infer the location.

5.3 Device classification

The devices are classified periodically, approximately every 15 minutes. For devices which have sent new data since the last classification, the classification is recalculated, if the confidence of the new class is higher as than the older one, it gets overwritten. The classification takes the stored domain information and SSDP data to identify the devices, using a previously trained and analysed model. Further Non-IoT devices are classified as such.

To prepare the collected data, the same aggregator as in the training part is used.

```
{"geo":{
        "lat":null,
        "lng":null,
        "city":"Seattle",
        "state":"Washington",
        "country":"United States of America",
        "stateCode":"WA",
        "postalCode":"98109-5210",
        "streetName":"Terry Avenue North",
        "subPremise":null,
        "countryCode":"US",
        "streetNumber":"410"
        },
"url":"https://www.amazon.com/",
"logo":"http://logo.bigpicture.io/logo/amazon.com",
"name":"Amazon",
[...]
"category":{
        "sector":"Technology",
        "industry":"Software & Computer Services",
        "subIndustry":"Internet",
        "industryGroup":"Technology"
        },
"foundedYear":1994,
"domainAliases":[...],
"emailProvider":false,
"id":"a8320cc9-d573-4b60-9af5-aefd300e8d88"}
```


Figure 5.2: Essential parts of the bigpicture.io lookup of Amazon

It aggregates the data hourly. One hour is a reasonable time window, due to the typically shorter time-to-life of DNS entries (See Section: 2.2). From the NTP requests is a time series built and the average time between such requests is used for the model to classify the device.

The confidence and the class is then also stored in the SQLite data, which can be accessed by the front end. Similarly, as the name of the device the user can overrule this classification. To prevent this manual classification from being overwritten, the confidence of such a classification is set to 100%.
For Hubs and Voice Command Devices, which often act as Hubs, additionally the classes of the managed devices, are needed. This information cannot be assessed by the machine learning model. Hence, the user must add this information manually to provide the correct privacy leak estimation.

5.4 Privacy leaks estimation

The leaks are visualised on the one hand by devices, on the other hand by companies. The privacy leaks are estimated based on the device class, as well as if the device is connecting to ad or tracking services and where the connection endpoints are located.

This module uses the collected and augmented data along with the results of the device classification process. The privacy implications are calculated, and the results and the privacy leaks are then visualized.

To provide real-time data, the same technique as implemented by the overview pages of the Pi-hole is applied. It uses a simple polling algorithm, in our case every minute the PHP APIs are queried, and the updates are displayed. This is achieved using the *setTimeout* function provided by JavaScript, which calls a function after a specific time. The timeout is set every time new data is received.

5.4.1 Devices

For each device, similarly to the long-term data graphics provided by the Pi-hole, the number of connections is shown on a stacked bar chart in real time and divided in blocked and unblocked traffic for a selected time scale.

Further the traces are displayed in a Sankey diagram. On the one side, the URLs of the DNS queries are shown and on the other end the countries the domain names are registered in. The thickness of the connection between a country and a domain is determined by the numbers of connections in the selected time span. Domains which are blocked by the Pi-hole and therefore rated as tracking or ad provider are coloured red.

Finally, the calculated privacy leaks are displayed in a table, similarly to the privacy nutrition label [39]. To provide an understanding on which basis the leaks are calculated, the assumed data types which are collected by the specific device are displayed, too.

5.4.2 Companies

In contrast to the device detail view, for companies there is no stacked bar chart provided, since there are most likely multiple devices responsible for sending data to the selected companies.

The Sankey diagram, shows the data flow from devices to a specific domain associated with the selected companies. Further correspondingly to the device detail view, the country in which the domain is registered is shown. The thickness of the connections is fixed by the number of DNS queries to the specific domain. Additionally, a combined privacy leak estimation for all selected companies is provided.

This estimation shows in the same manner, as the device view, the assumed data types which are used to calculate the privacy implications.

5.5 Pi-hole integration

To easily integrate our add-on into the Pi-hole an install script is provided, which takes care of the integration of the UI, in detail the PHP and JavaScript files are copied to the location where the correspondingly UI files of the Pi-hole are stored. The files of the Pi-hole remain untouched apart from the *header.php*. This file contains the navigation bar and clearly, to access the privacy leaks UI an additional menu item must be provided. This installation script also prepares the environment of the Raspberry Pi for the back end. It installs the required python version, the Redis server and tcpdump, which is necessary by scappy to intercept the traffic. To make the Raspberry Pi ready to run the python *scikit-learn* package, *libatlas-base-dev* and *python3-scipy* are installed.
Further it creates a working environment e.g., folders and a virtual python environment for the back end and copies the files to this location. Further the permissions of the database and the folders are set in such a way, that the front end, run by the Pi-hole user can read and write to the database. In this virtual environment, the required python packages are installed, mainly *scapy*, *redis* and *scikit-learn*. For the *scikit-learn* a specific version is required, to be compatible with the previously trained and stored machine learning model. In this case the version is set to *0.22.2.post1*.
To process the WHOIS results, *pycountry* and to query the web APIs, the package *requests* is used.
For the scheduling of the reclassification the *schedule* package is used and to transform the main python script into a Linux daemon, *python-daemon* is used.

5.6 Machine learning

In this section we describe step by step how the different machine learning models are trained and validated and in a second step how the best performing model is then prepared to be exported for the Pi-hole classification module.

5.6.1 Training & validation

To extract and aggregate the features provided by the test data, this data is beforehand filtered to speed up this recurring calculation. Therefore, the provided pcap-files are read using a python script and the package *scapy*. To speed up the filtration the tcpdump plugin is used by *scapy*. The applied filter is: *udp.port 53 || udp port 1900*, because DNS requests are UDP packets on port 53 and SSDP is using UDP port 1900. To minimize overhead, only the necessary information, namely the MAC address of the sender, the original time, as well as the time difference between packets, the type of the packet either DNS or SSDP, are used. For DNS packets the requested domain is stored and for SSDP the content of the packet is saved.

Further the csv-files contain approximately 24 hours of data, as the provided pcap files of the data set provided by the UNSW Sidney [16].
The pcap files of the data set from the IoTFinder [81] were split in five minutes per file, with one folder for each day. Therefore, these files were filtered and combined to roughly 24 hours per csv file using the same python script with some slight modifications to combine a folder to one file.

To aggregate the filtered data a python class is written, this aggregator provides time windows of one hour. This time span is reasonable, because it is bigger than the TTL of DNS records. Since there are time gaps between the two data set and within the IoTFinder data sets, the aggregator needs to take this in account when calculating the time intervals between arrival times of two consecutive packets. This calculation is needed to build a time series. This functionality uses a filter to select only NTP domain requests, to provide basic network behaviour. To be precise, domain names containing **.ntp.org* or **.nist.gov* or one of the following NTP servers are selected:

- time{1-4}?.google.com
- time{1-5}?.facebook.com
- time.windows.com
- time.apple.com
- time.euro.apple.com
- npt{1-5}.stratum{1-2}.ru
- ntp.ripe.net
- clock.isc.org
- ntp.isc.org
- ntp{1}?.time.nl

Moreover, a functionality to aggregate the collected data either by DNS requests or SSDP data is provided. Thereby same occurrences of DNS names or parts of the SSDP data are counted, to build a bag of words. The data of SSDP requests is split up in parts separated by whitespace, and then identifiers are filtered. All identifiers start with *http://*, therefore parts starting therewith are filtered.

To provide the machine learning methods from the *sklearn* packet, with data the aggregated data must be stored in a matrix with the dimensions $n\_samples \times n\_features$. To train the models the correct classes must be provided in an array with the length of $n\_samples$.

Therefore for each aggregated hour of DNS data, filtered SSDP data and the average of the NTP packet arrival times from a device, a row in the matrix is created. For each row all features must be provided, hence rows of the length *n_features* are created and initialized with 0. Then for each feature presented in the given data for the device, the corresponding column for the feature is set to the provided value.

The matrix is then used to create a train and test data split, the training split, with the default split of 75% training data and 25% test data, is then used to train the three different models (Random forest, K-Nearest neighbour, and multinomial Bayes). The splitting and the three machine learning algorithms are provided by *sklearn*. The test data split is used to validate the models. To cross validate the models the entire converted data is provided, since the k-cross validation splits the data itself in k parts. The methods to validate, respectively to cross validate, are also provided by *sklearn*. A coarse-grained pseudo code overview of the machine learning and validation algorithm can be seen in Algorithm 5.1.

To visualize the results, especially the heat maps, *matplotlib* is used. The basis of

Algorithm 5.1: ML-Training

```
   Input: A list of files files
 1 for f ∈ files do
 2 |   for row ∈ f do
 3 |   |   aggregator.add(row)
 4 |   end
 5 end
 6 for hour ← 0 to aggregator.hours do
 7 |   converter.addSamples(aggregator.data[hour])
 8 end
 9 data, classes = converter.convert()
10 X_train, X_test, y_train, y_test = trainTestSplit(data, classes)
11 for model ∈ rf, bayes, knn do
12 |   model.fit(X_train, y_train)
13 |   p = model.predict(X_test)
14 |   show report(y_test, p)
15 |   show crossValidationReport(model, data)
16 |   show heatmap(y_test, p)
17 end
```

this visualization is a normalized confusion matrix. It shows the percentage of correct and incorrect classified instances.

5.6.2 Pi-hole integration

Since the Raspberry Pi is only capable of running a 32-bit version of Python, the model must be trained using a 32-bit version of Python, to be compatible with the Raspberry Pi. Unfortunately, the 32-bit version has a memory limit of around 2 GB. For this reason, the data extraction and model learning must be split in two parts, because the data extraction alone exceeds the 2 GB memory limit.

As a result of this the data aggregation and extraction is performed in a 64-Bit version of Python and the intermediate results are stored in files. These results are a list of classes representing the according class of the sample matrix. Further a sorted list of features is provided, to be able to create test data. This is needed because the features and classes are number encoded. In contrast to the training and validation phase the sample matrix is not stored in a simple matrix, since this would exceed, as well the memory limits in the training phase. Therefore, the simple matrix is converted in a sparse matrix. A sparse matrix only stores values not equal to zero. Because in this case most of the values are zero i.e., there are more than 6.500 features for the different domains, the average NTP time interval and the parts of the SSDP protocol and many devices only connect to a few domains, resulting in many zero values.

In the next step the best performing machine learning algorithm is used to train a model. This model is then persisted using *joblib*. This is a lightweight library to persist large binary python objects. This library is further recommended by *sklearn*. The persisted model is then loaded at the start of the back end into the machine learning module.

To test the basic functionality of this persisted model, it is tested using three days of data, which are not used to train the model. For this purpose, the model is loaded and the parts of the sample set, similarly as at the train and test phase, are loaded and data from these parts are extracted, aggregated, and finally converted using the aggregator and the stored information which column of the matrix encodes which feature and how the classes are encoded.

5.7 Testing

To assure the functionality of the loaded machine learning model, first the accuracy of the model is tested. This is achieved by loading parts of the data set and storing it in the database which is accessed by the Pi-hole add on. This results in the approximately same accuracy as the tests of the machine learning part.

To test the integration of all components, parts of the test data are sent using *scappy* to the Raspberry Pi running the Pi-hole and the augmentation of it. The destination IP address of the DNS requests contained in the data sets are set to IP address of the

Raspberry Pi. Because the Pi-hole does not respond to unknown devices, beforehand an ARP response is created with the IP and MAC address of the device defined in the data set which sent the DNS request is created. This ARP response is then sent to the IP address of the Raspberry Pi. To create a valid ARP response the MAC address has to be the broadcast MAC address FF:FF:FF:FF:FF:FF.

The SSDP packages of the data set are sent respectively, but without an ARP response before, because the SSDP packages are only intercepted by our software and not the Pi-hole.

As the device classification uses the arrival times between NTP packages, each packet must be send with the same time difference as in the test data, although some deviations of the time differences are expected, because of several factors within the test network, mostly the varying round-trip time, which should be around 1-2 millisecond in the test network, because networks parts are connected via wires.

CHAPTER 6

Results

In this chapter the resulting privacy leak model is presented. Further the results of the machine learning experiments are shown. Lastly the technical probe is tested using parts of the datasets used to train and test the machine learning algorithms. Moreover, it is assessed if the technical probe reached all three design goals mentioned in Section 4.1.3.

6.1 Privacy leak model

Similarly, to the privacy-centred system model for connected homes by Bugeja et al. [37], a smart home H is defined as tuple $H = (D, C, R, L)$, where D are the IoT-devices, C are the companies receiving data from the smart home, R are the routes between the IoT-devices and their endpoints (companies) and L the potential leaks.

- D is a set of tuples (d_i, d_c) of device classes $d_c \in O$ and the identifier of the device d_i. This set is automatically assessed by the device classification component of the technical probe. A natural identifier d_i for a device is its MAC address.
- C is as well a set of tuples (c_i, c_l) describing the companies, which collect data from the smart home devices, where c_i is the name of the company and c_l the location where the company is based in.
- R the routes are the links of devices $d \in D$ and companies $c \in C$. Formally the set $R = \{(d, c) \mid d \text{ is sending data to } c\}$
- L is the set containing the potential privacy leaks, in this case the leaks are categorized by an alteration of the five types of privacy described in Section 2.3. This set is therefore defined as follows:
  $L = \{\text{Occupancy}, \text{State of body \& mind}, \text{Social life}, \text{Behaviour \& Action}, \text{Media}\}$

The device classes O containing the classes, which are contained in the ontology seen in Figure 3.1.
Further depending on the device class, specific types of data $t \in T$ are sent from such devices. This intermediate step helps to estimate the privacy leaks from multiple devices or which leaks may occur if different companies share data.

Therefore, we define a relation DT over $O \times T$ mapping device classes to the specific types of data, they are sending.
In Figure 6.1 the ontology is annotated with the according data, collected from devices belonging to the specific category. This annotated table is slightly modified from the original ontology. We merged some classes, which collect the same data. Furthermore, devices which are not connected to the internet are omitted, like Infrared controller and remote controls. Moreover, network devices like Wireless Signal Extenders are also not mentioned.
The shutter controller is merged with the Shades/ Blinds class, since the functionality and collected data is the same. The same argument applies for the different lights which are merged into one class in particular to the Lighting System class. Hubs are combined into one class, with no further distinction between open and closed ecosystems, because this fact has no implications on the resulting privacy leaks. As the lights, different audio speaker and music players are merged into the Audio Speaker/ Music Player class. Similarly, Floor Mopper, Floor Scrubber and Vacuum Cleaner are merged since such devices are often equipped with a combined functionality of these different classes and the collected data is very similar. Also, in our model there is no distinction between a pet door lock and a door lock. Finally, miscellaneous Portable Appliances the Dashboard and Personal Robot are omitted.
Additionally we added the Photo-Frame class, for devices which display photos or even synchronize them via cloud providers.
The abbreviations of types of data used in the modified ontology are defined in Table 6.1.

To calculate the privacy implication for one device the data types and therefore the device category is crucial. The privacy implications can consist of the five types of privacy described in Section 2.3, e.g., $PI(T, HU, AQ) =$ Occupancy. The derivation of types of data to privacy implications is described in the next subsection.

6.1.1 Privacy implications

The privacy implications are differentiated by a modified version of the five types of privacy: Occupancy, State of body & Mind, Social Life, Behaviour & Action and Media (See Section 2.3).

At least one of the following types of data is necessary to infer information about **Occupancy**. In the original five types of privacy, which are fitted for a smart city

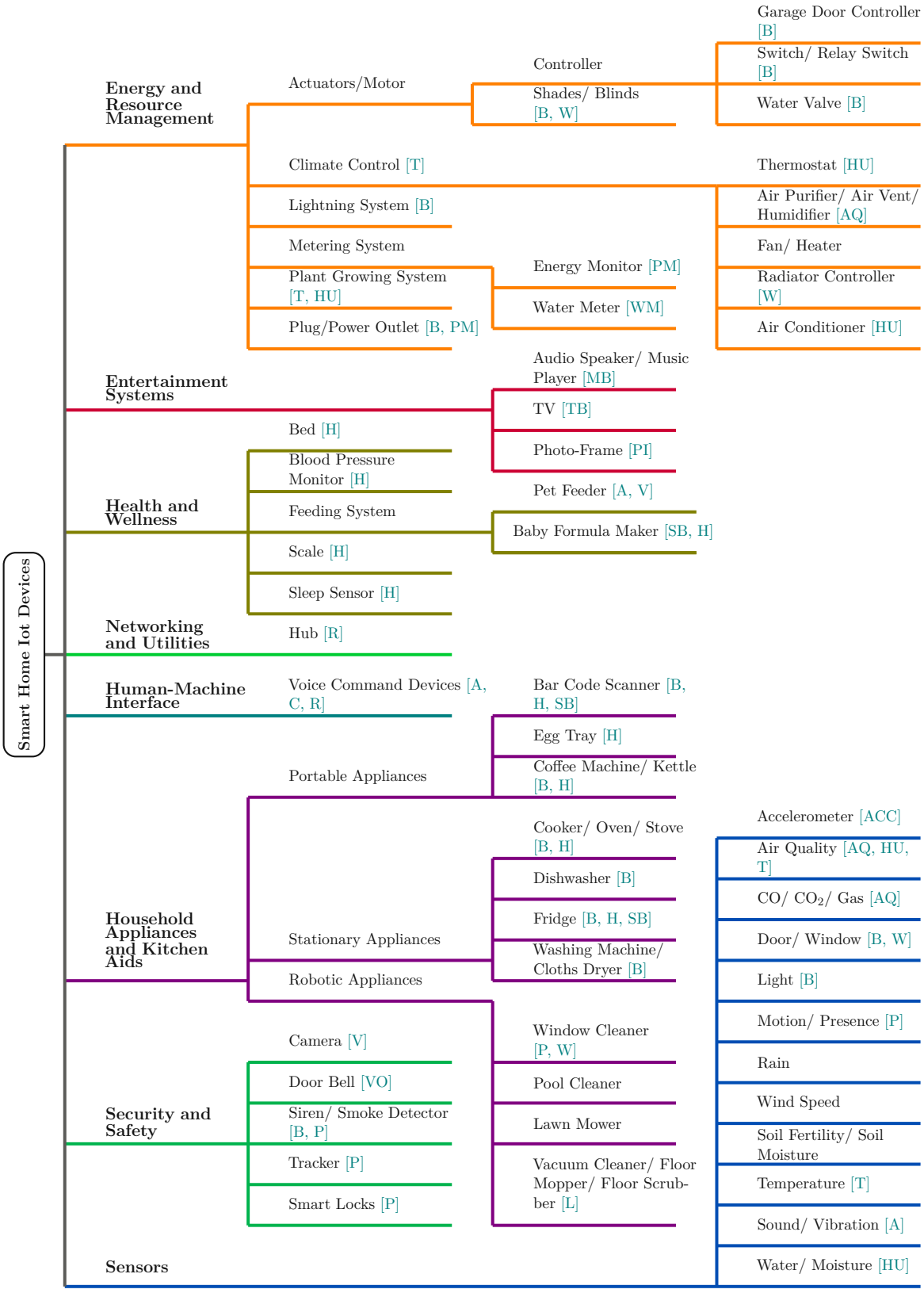

Figure 6.1: Smart home IoT devices data collection ontology. The characters in squared brackets, denote the data which is collected.

Type	Description
B	Binary Data
W	number of windows per room
T	temperature
HU	humidity
PM	Electrical power metering
WM	water metering
AQ	air quality related data, like CO_2 levels
MB	Music listening behaviour
TB	Television/ Streaming behaviour
H	Health related data
A	audio data
V	Video or pictures
VO	Video or pictures from outside the smart home
P	Presence
PI	Pictures
SB	shopping behaviour
L	layout of smart home
ACC	acceleration data
R	rooms and devices assigned to rooms
C	contacts and appointments

Table 6.1: Types of data collected by smart home IoT devices

environment this type is called location, but in this context the location is the smart home. Therefore, it is more useful to specify if the home is occupied or not. Clearly CO_2 levels and room clime data, to be more precise temperature and humidity are sufficient to infer occupancy of a room and therefore reveal a private location [52].
Further presence sensors clearly leak the occupancy of a room. Such sensors are typically built into Smoke Detectors.
Moreover, metering devices can detect occupancy, as well as Plugs and Power Outlets.
Obviously, cameras and microphones can also detect if a room is occupied.
Therefore, at least one of the data types P, PM, WM, AQ, A, V and the combination of T and HU are sufficient to detect the occupancy of a room, respectively leak information about the location.

Data containing information about the **State of Body & Mind** are, first of all, collected by devices which collect health related data, like Smart Health and Wellness devices. Smart Fridges can maintain grocery shopping lists and can therefore infer health related information.
Or in short by the data types: H, V, SB and A.

Information about **Social Life**, like interactions can be gathered through metadata of (voice/ video) chats, and observed conversations.
Further people can be identified via face or voice recognition.
Therefore the data types A, V, VO and C reveal information about social interactions.

Behaviour & Actions are induced by purchase patterns, as well as by an entertainment system.
Such data is derived by at least one of the types: MB, TB, SB. Further by one of following combinations: R and ACC administrated by the Hub, R and B which is also administrated by the Hub [85], as well as T & HU & AQ & L [52].

The privacy of **Media** is obviously implied by recorded audio and videos, like generated by Voice Command Devices, and Cameras. Hence, such information is clearly inferred by the data types A, PI, VO and V.

6.2 Classification Algorithm

In this section the accuracy of the three different machine learning algorithms is compared and analysed, using the model accuracy, precision, recall and F1-score, as well as the accuracy and the standard deviation of the k-cross validation. Moreover, for each algorithm a confusion matrix is provided.

The test datasets contain 16.356 DNS and SSDP hourly aggregated requests and is very imbalanced. For instance, from this hourly aggregated data only four examples are from the Blood Pressure meter class, the Smoke Detector/ Siren class is represented by 15 hours and the Vacuum Cleaner class is contained in 17 hours. On the other end, with the most representatives the camera devices have 3.123 hours of aggregated data and the Voice Command Device class provides 2.385 examples.

6.2.1 Random forest

The random forest model achieves an overall accuracy of 99,36% with a precision of 99,37%, with recognizing all classes with an accuracy over 99%, except the Non-IoT class, Switch/ Power outlet and the light bulb class, which is classified with at least 97% accuracy. The light bulb class is wrongly classified as Switch/Power Outlet with a probability of 1.8%. The misclassification as Switch/Power Outlet is most likely due to different devices of the same manufacturer present in both classes. This manufacturer uses the same domains to communicate with all devices in both classes.

Interestingly the power outlet is 1.4% wrong classified as Non-IoT, 0.46% as Hub, and for the same reason as the last class as Light bulb 0.46% are falsely classified.

The Non-IoT class is most commonly wrong classifying as Audio Speaker (0.49%) and Cooker/Oven/Stove (0.82%). This is most likely because of the vastly different behaviours among the different Non-IoT devices. A further reason is that devices of all these classes are often connected to Google servers.

As seen in the confusion matrix (Figure 6.2), there are some very unlikely wrong

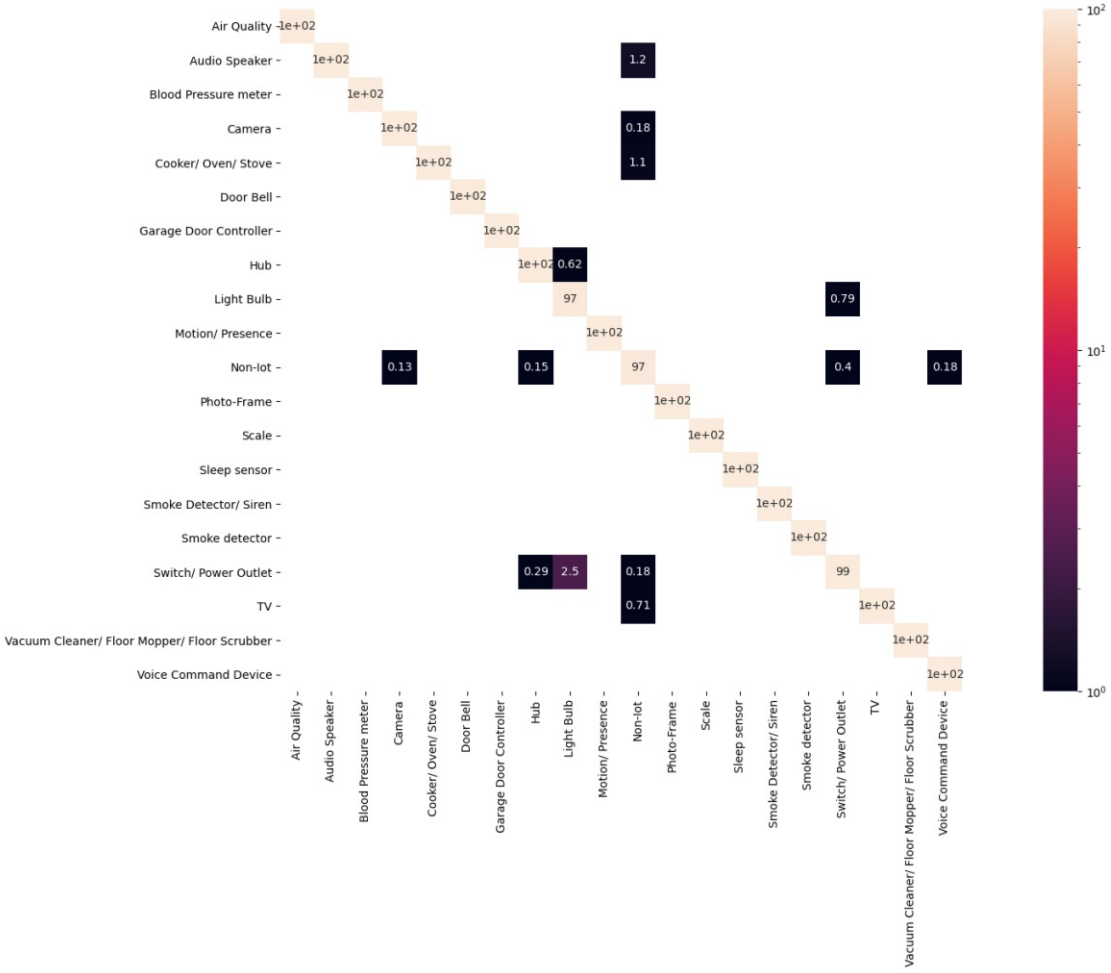

Figure 6.2: Confusion matrix shows the accuracy in percent of the device classes using the random forest classifier

classifications, additionally to the previously described hardly any instances of Cameras, Hubs and TV are wrongly classified, with the other classes reaching a perfect accuracy of 100%.

The 4 fold cross validation accuracy is 94% with a standard deviation of 0.08. Because of only four examples representing the Blood Pressure meter, there are not more

than 4 folds possible.

6.2.2 Multinomial Bayes

This classification algorithm scores not as well as the previous one, with an accuracy of 57.0% and a precision of 83.12%. The accuracy of the 4-cross validation is 60.0%.

In contrast to the random forest model, in the confusion matrix (Figure 6.3) the

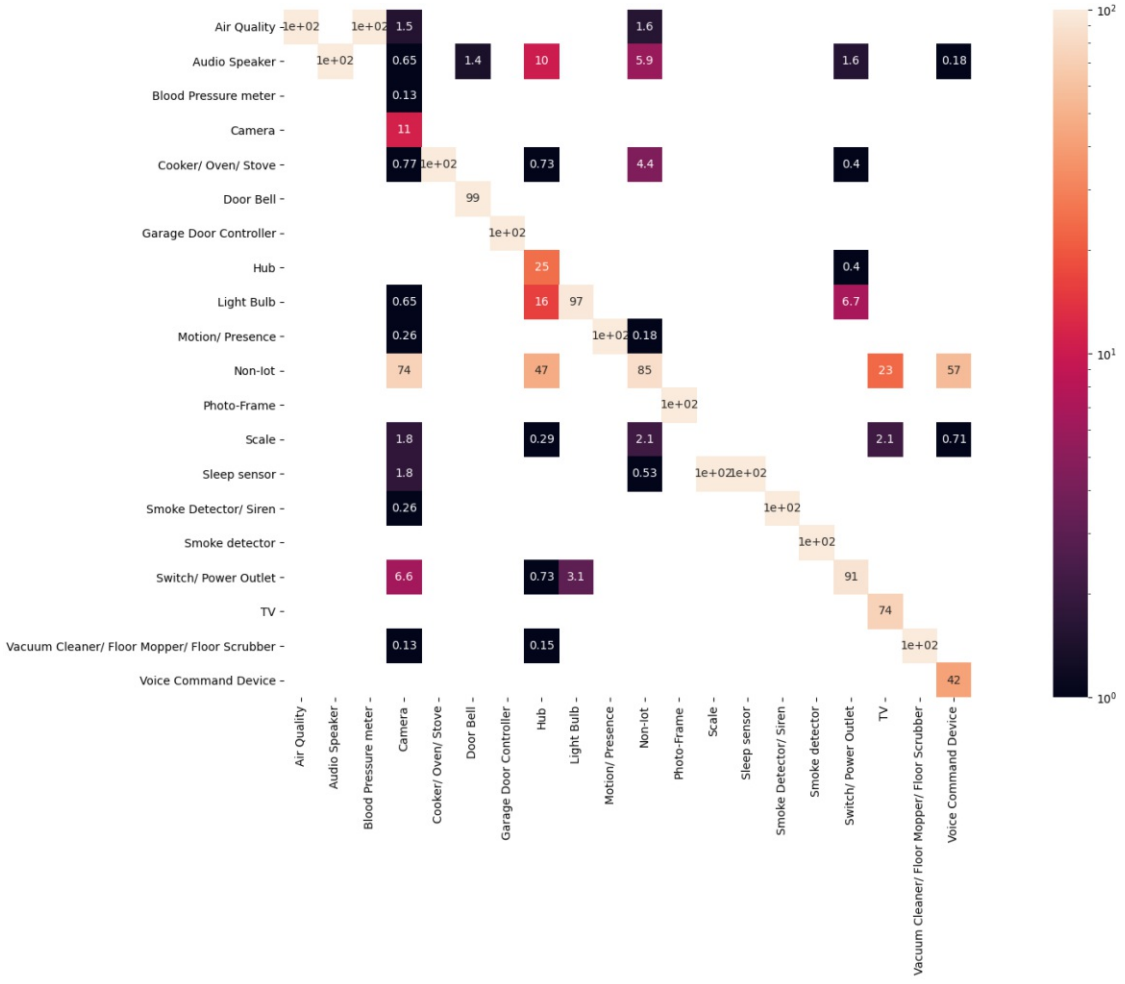

Figure 6.3: Confusion matrix shows the accuracy in percent of the device classes using the multinomial Bayes classifier

inferior performance can be seen easily. Two classes are not even classified once in the correct class, namely the Blood pressure meter and the Scale. Additionally, three classes are classified with an accuracy under 50% (Camera, Hub, Voice Command Device) and

three more with an accuracy under 98% (Non-IoT, Smoke Detector/ Siren, Switch/ Power Outlet).

6.2.3 K-nearest neighbour

The accuracy of this model is nearly as good as the random forest model, with 98.21% accuracy and a precision of 98.19%. The results of the 4-cross validations are: 91% accuracy with a standard deviation of 0.09. The confusion matrix for this model is provided in Figure 6.4.

Despite the good performance, the Blood Pressure Meter class is never classified correctly, instead the one device in this class is incorrectly classified as Camera.

There are only three more classes with an accuracy below 99%, in particular the Light Bulb class (96%), Non-IoT devices (94%) and Switch/ Power Outlets (97%).

6.2.4 Analysis

To summarize the results of the three different approaches, the measurements of all models are presented in a table (See table 6.2) and then compared. Further a table for each device class the performance and the incorrectly classified classes of all three models is provided (See table 6.3).

As can be clearly seen in this table the random forest outperforms the other classifiers in all classes, except the Hub class. This class is marginally more precise classified by the K-Nearest neighbour classifier with 99.72% in comparison to 99.58% of the random forest classifier.

Measurement	Random forest	K-Nearest neighbour	multinomial Bayes
accuracy	99.36%	98.21%	57.0%
precision	99.37%	98.19%	83.12%
recall	99.36%	98.21%	57.0%
F1-score	99.37%	98.19%	55.0%
4-cross validation (std)	94% (0.08)	91% (0.09)	60.0% (0.02)

Table 6.2: Results of the machine learning algorithms

Class	Random forest		K-Nearest neighbour		multinomial Bayes	
	Correctly classified	Incorrectly classified	Correctly classified	Incorrectly classified	Correctly classified	Incorrectly classified
Air Quality	100%		100%		100%	
Audio Speaker	100%		100%		100%	

Table 6.3 (continued)

Class	Random forest		K-Nearest neighbour		multinomial Bayes	
	Correctly classified	Incorrectly classified	Correctly classified	Incorrectly classified	Correctly classified	Incorrectly classified
Blood Pressure meter	100%		0%	Camera (100%)	0%	Air Quality (100%)
Camera	99.73%	Non-Iot (0.27%)	98.5%	Cooker/ Oven/ Stove(0.14%) Hub (0.14%) Light Bulb(0.68%) Non-Iot (0.27%) Smoke Detector/ Siren (0.27%)	10.08%	Air Quality (1.5%) Non-Iot (77%) Switch Power Outlet (5.3%) Scale (1.6%) Sleep Sensor(1.8%) 6 Others (2.72%)
Cooker/ Oven/ Stove	100%		100%		100%	
Door Bell	100%		98.6%	Camera (1.4%)	99.3%	Audio Speaker (0.7%)
Garage Door Controller	100%		100%		100%	
Hub	99.58%	Light Bulb(0.27%) Non-Iot (0.14%) Switch Power Outlet (0.14%)	99.72%	Light Bulb(0.14%) Non-Iot (0.14%)	22.49%	Audio Speaker(11%) Light Bulb (17%) Non-Iot (48%) 2 Others (1.51%)
Light Bulb	98.2%	Switch Power Outlet (1.8%)	95.8%	Hub (1.8%) Switch Power Outlet (2.4%)	97.6%	Switch Power Outlet (2.4%)
Motion/ Presence	100%		100%		99.06%	Switch Power Outlet (0.94%)
Non-Iot	98.04%	Audio-Speaker (0.49%) Cooker/ Oven/ Stove(0.82%) Switch Power Outlet (0.16%) TV (0.49%)	93.55%	Audio-Speaker (0.66%) Camera (1.8%) Cooker/ Oven/ Stove(0.99%) Scale (3.3%) 4 Others (1.8%)	84.67%	Audio-Speaker (6.6%) Cooker/ Oven/ Stove(3.8%) Hub (1.2%) 5 Others (1.63%)
Photo-Frame	100%		100%		100%	
Scale	100%		100%		0%	Sleep Sensor (100%)

Table 6.3 (continued)

Class	Random forest		K-Nearest neighbour		multinomial Bayes	
	Correctly classified	Incorrectly classified	Correctly classified	Incorrectly classified	Correctly classified	Incorrectly classified
Switch/ Power Outlet	97.68%	Hub (0.46%) Light Bulb (0.46%) Non-Iot (0.14%)	96.76%	Camera (0.92%) Hub (0.46%) Motion/ Presence (0.46%) Non-Iot (1.4%)	88.9%	Audio Speaker(2.8%) Cooker/ Oven/ Stove(1.8%) Motion/ Light Bulb (6.5%)
Sleep sensor	100%		100%		100%	
Smoke Detector/ Siren	100%		100%		88%	Non-Iot (12%)
TV	99.66%	Non-Iot (0.33%)	99.02%	Non-Iot (0.98%)	72.7%	Non-Iot (26%) Scale(1.3%)
Vacuum Cleaner/ FM/ FS	100%		100%		100%	
Voice Command Device	100%		100%		43.65%	Non-Iot (56%) Scale(0.35%)

Table 6.3: Precision of the machine learning algorithms

6.3 Technical probe

The technical probe is deployed on a Raspberry Pi, located in a home network. To assess the performance of the software, parts of the data sets, which are not used to train the classification model deployed on the raspberry pi, are sent to this device (See Section 5.7).

First, to assess the accuracy of the machine learning classifier built into the technical probe, approximately two hours each of two distinct test data are sent to the Raspberry Pi. After this time period, we compared the classification of the software with the ground truth. Most of the 38 simulated devices are classified with high confidence. Most of the devices are moreover correctly classified, which by absolute numbers is 35 out of the 38 devices.

One wrongly classified device is a Belkin device, is labelled as Power Plug instead of Motion/ Presence. This discrepancy is most likely due to the similarities between the Motion/ Presence sensor and the Power Plugs of the same vendor. In fact, in the used

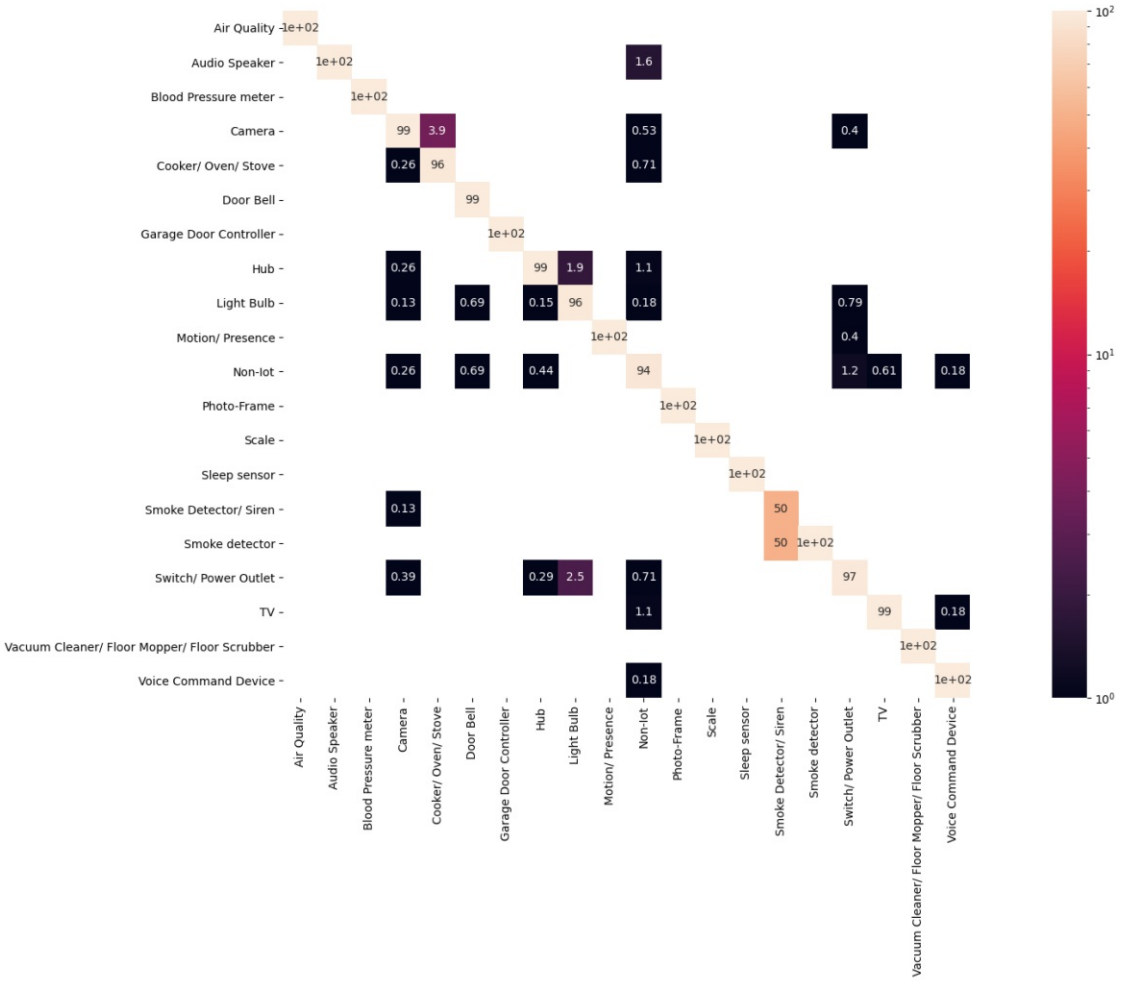

Figure 6.4: Confusion matrix shows the accuracy in percent of the device classes using the K-nearest neighbour classifier

parts of the data set, the motion device connects to a subset of the plug. The UPnP data these devices send is also identical.
For the same reason, the model classifies devices from Withing wrong. The Withing sleep sensor and camera are classified as Scale. The scale class only consists of one device developed by Withing.

In the next part we describe how the three design goals of privacy empowering tools proposed by Seymour et al. [5] are reached.

6.3.1 DG1: Legibility

To provide transparency to which endpoint each device connects, the intercepted traffic is shown in real-time and historical data of it is provided.
A more coarse-grained functionality is already provided by the Pi-hole. It shows live and historical data for the entire network. Therefore, for this design goal, a resembling look and feel is used, in fact many components provided by Pi-hole are reused, like the data rage picker and a tabular overview of the blocked and not blocked traffic. This overview only shows how much DNS requests are sent in the specified time range and how much of it is blocked by the Pi-hole.

The connection endpoints and amount of DNS requests for each endpoint are shown in a Sankey diagram, where each endpoint is connected to the corresponding company owning the domain and each company is connected to the country, where the company is located. To show users not only how much traffic is blocked, but also which domain is blocked, the blocked domains are coloured red in the Sankey diagram (See Figure 6.5[1]).

Furthermore, the same analysis is also provided for companies. This analysis does not

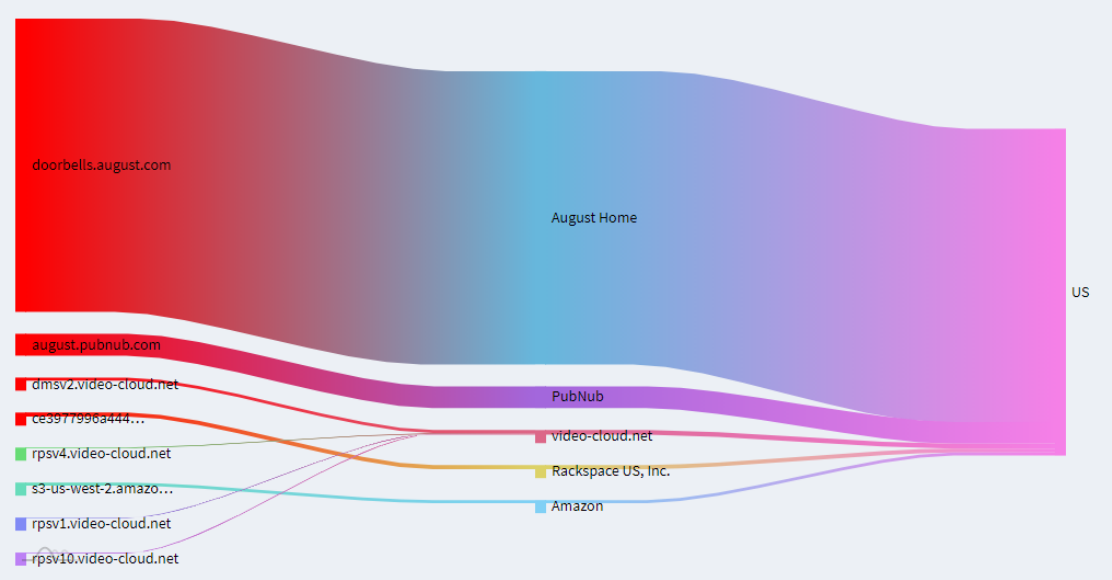

Figure 6.5: Connections of an August doorbell[1]

show the tabular overview, instead solely the Sankey diagram is displayed. This diagram additionally provides the information, which device is connected to which endpoint.

[1]We chose this device randomly and do not intend to pin-point any of the companies, but we are trying to highlight a general issue

6.3.2 DG2: Interpretability

To improve the interpretability of the data, which is sent from smart home devices, an estimation of which privacy leaks may occur, is provided. The leaks are calculated using the previously introduced privacy leak model (See Section 6.1) and displayed in a table. Similarly to the so called "nutrition label for privacy" [39, 40], all potential privacy leak categories are displayed all the time. Privacy leaks which are not occurring are shown greyed out, otherwise they are displayed black.
This nutrition label is displayed as a tabular, where the rows are the privacy leaks and the columns are devices respectively companies if more than one company is selected. A cell is filled with a white exclamation mark and red background if the corresponding device is responsible for the privacy leak in the corresponding row.

For a single device additionally the assumed types of data, which are typically collected by the device category of the selected device, are shown and described. This privacy label for an August doorbell[2] can be seen in Figure 6.6.

For a single company for each device a column with its privacy leaks is added. The class

Figure 6.6: Privacy leaks of an August doorbell[2]

of the devices is also displayed in the header of the column. Further an additional column is added for the combined privacy leaks, which are not always solely the combination of the privacy leaks of all devices. Certain combinations can yield further privacy leaks.

If multiple companies are selected instead of all devices responsible for the privacy leaks, each selected company and its privacy leaks are shown. Similarly, as earlier, a combination of the privacy leaks of all devices of the companies is shown in an additional column.

[2]We chose this device randomly and do not intend to pin-point any of the companies, but we are trying to highlight a general issue

For the companies August and Philips[3] this is shown in Figure 6.7.

Privacy leaks

types of privacy	August Home	Philips Hue	Combination
State of body & Mind	-	!	!
Occupancy	-	!	!
Social Life	!	!	!
Behaviour & Action	-	-	-
Media	!	!	!

Figure 6.7: Privacy leaks of the two companies August and Philips [3]

6.3.3 DG3: Actionable Choices

Unwanted traffic can be blocked for each device by creating an requisite list. This functionality is already provided by the Pi-hole. Moreover, the Pi-hole provides block lists created by experts, which block ad content providers and trackers for all devices by default. These lists can be altered respectively created and assigned to further filter the traffic for a device or a group of devices.
The information how much traffic and the traffic to which domain is then, as mentioned before, displayed in the tabular form and the Sankey diagram.

[3]We chose these companies randomly and do not intend to pin-point any of the companies, but we are trying to highlight a general issue

CHAPTER 7

Discussion

In this chapter we discuss the results of our machine learning method for classifying internet of things smart home devices, by their network traces. Further, we examine the results of how a novel model for estimating privacy leaks in smart home environments is developed. In addition to the technical probe, incorporating both previously mentioned results is discussed and compared to similar probes. Finally, the limitations of our approach are described.

7.1 Device classification

Despite the limited feature set (DNS and SSDP) and fine-grained classes, our machine learning approach to classify smart home internet of things devices, proved to have the same accuracy range as other State-of-the-art approaches. We reached an accuracy of 99.36%, with the worst class classified at least with 97.68% accuracy. Other State-of-the-art approaches used to achieve such an accuracy level typically require more features from the network traffic and fewer classes.

In fact, compared with the approach taken by Sivanathan et al. [16], which classifies parts of the same data set, we used 99.88 % accuracy using 15 different features, including statistic about the flow characteristics, like the amount of data transferred. Their approach provides using just domain name attributes with only 79.48 % accuracy.

Unfortunately, comparing our results with the results of the method to classify devices using the other part of the data set, is not possible since they do not provide an accuracy measurement for their approach [18]. Nonetheless, their approach does not of

distinguish between devices of the same manufacturer [18]. Our method also shows this behaviour, but to a much smaller extent.

7.2 Privacy leak model

We enhanced an existing privacy centred formal model of smart homes, to be able to infer privacy leaks. The classification of the leaks is based on the five types of privacy, which are optimized for categorizing privacy leaks occurring in smart cities [27]. Therefore, we modified these five types of privacy slightly, to better fit a smart home environment. The only major change is the change of the **Location** to **Occupancy**, because in the smart home context the occupation of rooms respectively the smart home is more insightful than the specification which type of location e.g., workplace, home it is.

7.3 Privacy empowering tool

The privacy leak model and the device classification machine learning model are used as backbone for the technical probe, which is based on the Pi-hole. Using these models, it is possible to reach the three design goals: Legibility, Interpretability and Actionable Choices (See section 4.1.3).

The last goal, Actionable Choices, is obviously already met by the Pi-hole's capability to block specific traffic from selected devices. Moreover, it provides lists of ad content providers and trackers collected by experts, to automatically block such traffic. To show users which parts of the traffic is blocked, blocked domains are displayed in red.

Legibility is provided by showing the traffic for each smart home device in real time and as historical data. To provide even more insight for the users, the connection endpoints are analysed and the respective company owning the endpoint and the country they are located in is shown.

The design goal interpretability is reached through a novel approach incorporating the previously mentioned privacy leak model. Our strategy is to provide an estimate of which information is collected. In detail, aspects of privacy, which can be derived from the data sent by smart home internet of things devices, are shown. Despite other related work, which focus more on educating users how data is sent, respectively stored at which companies and which ad or tracking provider is contacted, this approach tries to help users understand how their data can be used and potentially abused. Further our model can provide information how data sharing between companies can impact privacy, because if two companies share specific data, they can yield further information, resulting in more privacy leaks.

7.3.1 Related privacy empowering tools

In this section we compare our approach to improve privacy awareness, with approaches from similar privacy empowering tools. These tools are Aretha, HomeSnitch and IoTInspector.

Similar to our tool, Aretha [5] displays the intercepted traffic as a time series as well as the geographic distribution of destinations. Further they provide a total overview which device transmits to which company. In contrast they provide these overviews in respect to the amount of transmitted data, we provide these overviews proportional to the number of DNS requests.
But the major distinction between these two privacy empowering tools is how they help users understand what the privacy implications are. Aretha on the one hand provides educational information how data is transmitted and stored, we on the other hand show which privacy implications can occur. We also provide additional insight into what data companies receive from the devices connected to them.

In comparison, HomeSnitch's [10] primary goal is to classify the behaviour of smart home devices and provide policies, which block unwanted behaviour. Like our approach, they use machine learning models to classify devices, but they classify the devices by their behaviour not their functional classes. Further HomeSnitch does not provide an overview to which companies a device is connecting. They provide for each a report containing which behaviour is sent to which domain. Our technical probe cannot distinguish which behaviour is sent to a specific domain, but we analyse which company owns the domain and which information can be inferred from a functional device class in general.

IoTInspector's [7] goal is to crowdsource a big labelled smart home network data set. In contrast to our technical probe, IoTInspector cannot classify devices automatically. Therefore, the users must provide the classification (labels). Moreover, in contrast to our tool, their tool provides only rudimentary network flow statistics and does not analyse which company owns which domain. Additionally, this tool lacks an analysis about potential privacy leaks as well as an overview of the contacted companies or domains.

7.4 Limitations

Since not all devices in a functional device class are equipped with the same types of sensors, this model is only an estimation, which data such devices potentially transmitting. Further because we are only capture the DNS requests and SSDP data, our technical probe cannot differentiate which company is receiving which amount of data, but we can show how often in respect to the other contacted companies a company is contacted. However, with our setup it is not possible to evaluate, if the companies are receiving the data we assume, further we cannot evaluate if companies derive the same information as we suppose. This can be achieved using automated Man-in-the-middle attacks, but this

is out of our scope.

Moreover, a known limitation of machine learning models, like we used, is that they do not provide the same accuracy with unknown devices. Further our machine learning model, cannot infer which types of devices are handled by hubs or voice command devices, which act as hubs, therefore the user must add these categories manually.

CHAPTER 8

Conclusion

In this thesis we developed a technical probe to visualize and estimate privacy leaks in smart homes, based on the network wide ad-blocker Pi-hole. As other tools, we visualize the connections of each connected device. However, despite most other privacy empowering tools, our tool further estimates which data might leak from the smart home. Our classification of the privacy leaks is based on the five types of privacy [27], which we optimized to fit the smart home context. These leaks are then represented, similarly to the nutrition facts, in tabular form.
Since current research shows a lack in estimating data leaks from smart homes, only results of specific devices are available. Therefore, we developed a novel model capable of inferring such leaks from functional device classes. The functional device classes are based on an ontology containing 69 different classes. Also, this model even allows to infer what information companies can further derive if they share data with each other. This is achieved through an intermediate representation, so called data types. We use 20 different data types, which are collected by specific devices, to infer the privacy leaks.

To provide the functional device classes as input for this model, we created a machine learning model, which can reliably classify smart home devices, by the limited network features collected by the Pi-hole. Therefore, to identify device types we compared three different machine learning models, namely Random forest, K-nearest neighbour, and Multinomial Bayes, to categorize smart home internet of things devices, by their DNS traffic and SSDP messages. The Random forest performed best, resulting in an overall accuracy of 99.36%. This model deployed in the technical probe identified 35 out of 38 devices correct after two hours.

8.1 Future Work

The current privacy leak model is backed by only a few results which measure the impact of combining data from different smart home internet of things devices to infer behaviour, activities, occupation, or other personal information. Therefore, some combinations of data types are yet not backed by research. Additional research to cover new combinations of data types and incorporating new data types, like shown in [52], can improve the accuracy of our model.

Furthermore, to provide more insight in how data is processed by the companies, the model could also take into account the privacy notices of the companies respectively the devices. To provide a quick intuition of this notice natural language processing could be used and the results could then be displayed in a similar tabular way.

Acronyms

API Application Programming Interface. 5, 36, 38, 39

ARP Address Resolution Protocol. 3, 4, 17, 18, 43

Bluetooth LE Bluetooth Low Energy. 5

DNS Domain Name System. 2–7, 18, 19, 26, 34, 35, 37–43, 49, 56, 59, 61, 63

GDPR General Data Protection Regulation. 11, 35

HbbTV Hybrid Broadcast Broadband TV. 20

HCI Human–computer interaction. 8

HTTP Hypertext Transfer Protocol. 5, 19, 26

HTTPS Hypertext Transfer Protocol Secure. 18

HVAC Heating, ventilation, and air conditioning. 17

ICMP Internet Control Message Protocol. 18

IoT Internet of Things. 1, 5, 7, 11, 15, 24, 29, 31–33, 36, 47–50, 52

IP Internet Protocol. 4, 17–19, 30, 34–36, 42, 43

ISP Internet Service Provider. 11, 18

MAC Media Access Control. 4, 5, 17, 18, 30, 31, 39, 43, 45

mDNS Multicast DNS. 2

NDP Neighbour Discovery Protocol. 18

NTP Network Time Protocol. 4, 6, 7, 37, 40–43

OUI Organizationally Unique Identifier. 4, 5

RR Resource Record. 19

SEND Secure Neighbour Discovery. 18

SSDP Simple Service Discovery Protocol. 4, 18, 19, 34–36, 39–43, 49, 59, 61, 63

TCP Transmission Control Protocol. 18

TLS Transport Layer Security. 4

TTL Time to live. 7, 40

UDP User Datagram Protocol. 19, 39

UPnP Universal Plug and Play. 2, 19, 55

Bibliography

[1] Sam Lucero et al. Iot platforms: enabling the internet of things. 2016.

[2] James Manyika and Michael Chui. By 2025, internet of things applications could have $11 trillion impact. *Insight Publications*, 2015.

[3] Nina Gerber, Benjamin Reinheimer, and Melanie Volkamer. Home sweet home? investigating users' awareness of smart home privacy threats. In *Proceedings of An Interactive Workshop on the Human aspects of Smarthome Security and Privacy (WSSP), Baltimore, MD, August 12, 2018*. USENIX, Berkeley, CA, 2018.

[4] Peter Tolmie and Andy Crabtree. Deploying research technology in the home. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, CSCW '08, page 639–648, New York, NY, USA, 2008. Association for Computing Machinery.

[5] William Seymour, Martin J. Kraemer, Reuben Binns, and Max Van Kleek. Informing the design of privacy-empowering tools for the connected home. *ArXiv*, abs/2001.09077, 2020.

[6] Max Van Kleek, W. Seymour, R. Binns, Jun Zhao, D. Karandikar, and N. Shadbolt. Iot refine: Making smart home devices accountable for their data harvesting practices. pages 9 (6 pp.)–9 (6 pp.), 01 2019.

[7] Danny Yuxing Huang, Noah Apthorpe, Gunes Acar, Frank Li, and Nick Feamster. Iot inspector: Crowdsourcing labeled network traffic from smart home devices at scale, 2019.

[8] *Pi-hole®: Network-wide Ad Blocking - A black hole for Internet advertisements*, 2020 (accessed 20.05.2021). `https://pi-hole.net`.

[9] Timothy Libert. Exposing the hidden web: An analysis of third-party http requests on 1 million websites, 2015.

[10] TJ OConnor, Reham Mohamed, Markus Miettinen, William Enck, Bradley Reaves, and Ahmad-Reza Sadeghi. Homesnitch: Behavior transparency and control for smart home iot devices. In *Proceedings of the 12th Conference on Security and Privacy in*

Wireless and Mobile Networks, WiSec '19, page 128–138, New York, NY, USA, 2019. Association for Computing Machinery.

[11] *Fingerbank | Device Fingerprinting*, 2020 (accessed 20.05.2021). `https://www.fingerbank.org`.

[12] Noah J. Apthorpe, Dillon Reisman, Srikanth Sundaresan, Arvind Narayanan, and Nick Feamster. Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic. *CoRR*, abs/1708.05044, 2017.

[13] Abbas Acar, Hossein Fereidooni, Tigist Abera, Amit Kumar Sikder, Markus Miettinen, Hidayet Aksu, Mauro Conti, Ahmad-Reza Sadeghi, and A. Selcuk Uluagac. Peek-a-boo: I see your smart home activities, even encrypted! *CoRR*, abs/1808.02741, 2018.

[14] Ola Salman, Imad H. Elhajj, Ayman Kayssi, and Ali Chehab. A review on machine learning–based approaches for internet traffic classification. *Annals of Telecommunications*, June 2020.

[15] L. Bai, L. Yao, S. S. Kanhere, X. Wang, and Z. Yang. Automatic device classification from network traffic streams of internet of things. In *2018 IEEE 43rd Conference on Local Computer Networks (LCN)*, pages 1–9, 2018.

[16] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman. Classifying iot devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing*, 18(8):1745–1759, 2019.

[17] Andrew McCallum, Kamal Nigam, et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer, 1998.

[18] Iotfinder: Efficient large-scale identification of iot devices via passive dns traffic analysis.

[19] Rachel L. Finn, David Wright, and Michael Friedewald. *Seven Types of Privacy*, pages 3–32. Springer Netherlands, Dordrecht, 2013.

[20] Helen Nissenbaum. Privacy as contextual integrity. *Washington Law Review*, 79(1):119–157, February 2004.

[21] Jason I. Hong, Jennifer D. Ng, Scott Lederer, and James A. Landay. Privacy risk models for designing privacy-sensitive ubiquitous computing systems. In *Proceedings of the 5th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques*, DIS '04, page 91–100, New York, NY, USA, 2004. Association for Computing Machinery.

[22] Gary Marx. *Identity and anonymity: Some conceptual distinctions and issues for research*. Princeton University Press, 2001.

[23] N. Davies and M. Langheinrich. Privacy by design [from the editor in chief]. *IEEE Pervasive Computing*, 12(2):2–4, 2013.

[24] Louise Barkhuus. The mismeasurement of privacy: Using contextual integrity to reconsider privacy in hci. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, page 367–376, New York, NY, USA, 2012. Association for Computing Machinery.

[25] Leysia Palen and Paul Dourish. Unpacking "privacy" for a networked world. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, page 129–136, New York, NY, USA, 2003. Association for Computing Machinery.

[26] Roger Clarke. What's' privacy'. In *Australian law reform commission workshop*, volume 28, 2006.

[27] D. Eckhoff and I. Wagner. Privacy in the smart city—applications, technologies, challenges, and solutions. *IEEE Communications Surveys Tutorials*, 20(1):489–516, 2018.

[28] Patricia Smith Churchland. *Brain-wise: Studies in neurophilosophy*. MIT press, 2002.

[29] 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*, IMC '19, page 267–279, New York, NY, USA, 2019. Association for Computing Machinery.

[30] Vijay Srinivasan, John Stankovic, and Kamin Whitehouse. Protecting your daily in-home activity information from a wireless snooping attack. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, UbiComp '08, page 202–211, New York, NY, USA, 2008. Association for Computing Machinery.

[31] C. Debes, A. Merentitis, S. Sukhanov, M. Niessen, N. Frangiadakis, and A. Bauer. Monitoring activities of daily living in smart homes: Understanding human behavior. *IEEE Signal Processing Magazine*, 33(2):81–94, 2016.

[32] Natã M. Barbosa, Joon S. Park, Yaxing Yao, and Yang Wang. "what if?" predicting individual users' smart home privacy preferences and their changes. *Proceedings on Privacy Enhancing Technologies*, 2019(4):211 – 231, 2019.

[33] Daniel Bastos, Fabio Giubilo, Mark Shackleton, and Fadi El-Moussa. Gdpr privacy implications for the internet of things. In *4th Annual IoT Security Foundation Conference, London, UK*, 2018.

[34] European Union. Charter of fundamental rights of the european union. *Official Journal of the European Union*, 83(02), 2010.

[35] Jürgen Meyer, Norbert Bernsdorff, and Jürgen Meyer. *Charta der Grundrechte der Europäischen Union*. Nomos, 2006.

[36] Homin Park, Taejoon Park, Sang Hyuk Son, et al. A comparative study of privacy protection methods for smart home environments. *Int. J. Smart Home*, 7:85–94, 2013.

[37] Joseph Bugeja, Andreas Jacobsson, and Paul Davidsson. A privacy-centered system model for smart connected homes. In *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 1–4. IEEE, 2020.

[38] Jan Henrik Ziegeldorf, Oscar Garcia Morchon, and Klaus Wehrle. Privacy in the internet of things: threats and challenges. *Security and Communication Networks*, 7(12):2728–2742, 2014.

[39] Patrick Gage Kelley, Joanna Bresee, Lorrie Faith Cranor, and Robert W. Reeder. A "nutrition label" for privacy. In *Proceedings of the 5th Symposium on Usable Privacy and Security*, SOUPS '09, New York, NY, USA, 2009. Association for Computing Machinery.

[40] Patrick Gage Kelley, Lucian Cesca, Joanna Bresee, and Lorrie Faith Cranor. Standardizing privacy notices: An online study of the nutrition label approach. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, page 1573–1582, New York, NY, USA, 2010. Association for Computing Machinery.

[41] J. Bugeja, A. Jacobsson, and P. Davidsson. On privacy and security challenges in smart connected homes. In *2016 European Intelligence and Security Informatics Conference (EISIC)*, pages 172–175, 2016.

[42] J. Bugeja, P. Davidsson, and A. Jacobsson. Functional classification and quantitative analysis of smart connected home devices. In *2018 Global Internet of Things Summit (GIoTS)*, pages 1–6, 2018.

[43] Peter Hamernik, Pavol Tanuska, et al. Classification of functions in smart home. *International Journal of Information and Education Technology*, 2(2):149, 2012.

[44] Benjamin K Sovacool and Dylan D Furszyfer Del Rio. Smart home technologies in europe: A critical review of concepts, benefits, risks and policies. *Renewable and Sustainable Energy Reviews*, 120:109663, 2020.

[45] David C. Plummer. Ethernet address resolution protocol: Or converting network protocol addresses to 48.bit ethernet address for transmission on ethernet hardware.

STD 37, RFC Editor, November 1982. http://www.rfc-editor.org/rfc/rfc826.txt.

[46] T. Narten, E. Nordmark, W. Simpson, and H. Soliman. Neighbor discovery for ip version 6 (ipv6). RFC 4861, RFC Editor, September 2007. http://www.rfc-editor.org/rfc/rfc4861.txt.

[47] G. Jinhua and X. Kejian. Arp spoofing detection algorithm using icmp protocol. In *2013 International Conference on Computer Communication and Informatics*, pages 1–6, 2013.

[48] Blagoj Nenovski and Pece Mitrevski. Real-world arp attacks and packet sniffing, detection and prevention on windows and android devices. 04 2015.

[49] T. Chomsiri. Sniffing packets on lan without arp spoofing. In *2008 Third International Conference on Convergence and Hybrid Information Technology*, volume 2, pages 472–477, 2008.

[50] Yousef Amar, Hamed Haddadi, Richard Mortier, Anthony Brown, James Colley, and Andy Crabtree. An analysis of home iot network traffic and behaviour, 2018.

[51] K. Xu, F. Wang, S. Jimenez, A. Lamontagne, J. Cummings, and M. Hoikka. Characterizing dns behaviors of internet of things in edge networks. *IEEE Internet of Things Journal*, 7(9):7991–7998, 2020.

[52] Philipp Morgner, Christian Müller, Matthias Ring, Björn Eskofier, Christian Riess, Frederik Armknecht, and Zinaida Benenson. Privacy implications of room climate data. In *European Symposium on Research in Computer Security*, pages 324–343. Springer, 2017.

[53] Bogdan Copos, Karl Levitt, Matt Bishop, and Jeff Rowe. Is anybody home? inferring activity from smart home network traffic. In *2016 IEEE Security and Privacy Workshops (SPW)*, pages 245–251. IEEE, 2016.

[54] E. Furey and J. Blue. She knows too much – voice command devices and privacy. In *2018 29th Irish Signals and Systems Conference (ISSC)*, pages 1–6, 2018.

[55] Kristina Irion and Natali Helberger. Smart tv and the online media sector: User privacy in view of changing market realities. *Telecommunications Policy*, 41(3):170 – 184, 2017.

[56] M. Keshavarz and M. Anwar. Towards improving privacy control for smart homes: A privacy decision framework. In *2018 16th Annual Conference on Privacy, Security and Trust (PST)*, pages 1–3, 2018.

[57] D. C. Yacchirema, D. Sarabia-JáCome, C. E. Palau, and M. Esteve. A smart system for sleep monitoring by integrating iot with big data analytics. *IEEE Access*, 6:35988–36001, 2018.

[58] W. Wu, K. Cheng, and P. Lin. A remote pet feeder control system via mqtt protocol. In *2018 IEEE International Conference on Applied System Invention (ICASI)*, pages 487–489, 2018.

[59] J. Jung, C. Ji, J. Sohn, H. Meng, and B. Hwang. Nuripet: A smart pet feeding machine for sns. In *2016 IEEE International Conference on Consumer Electronics (ICCE)*, pages 117–118, 2016.

[60] Gierad Laput, Yang Zhang, and Chris Harrison. Synthetic sensors: Towards general-purpose sensing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, page 3986–3999, New York, NY, USA, 2017. Association for Computing Machinery.

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

[62] J. Li, Z. Li, G. Tyson, and G. Xie. Your privilege gives your privacy away: An analysis of a home security camera service. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pages 387–396, 2020.

[63] Suhuai Luo, Jesse S Jin, Jiaming Li, et al. A smart fridge with an ability to enhance health and enable better nutrition. *International Journal of Multimedia and Ubiquitous Engineering*, 4(2):69–80, 2009.

[64] Yaxing Yao, Justin Reed Basdeo, Smirity Kaushik, and Yang Wang. Defending my castle: A co-design study of privacy mechanisms for smart homes. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, page 1–12, New York, NY, USA, 2019. Association for Computing Machinery.

[65] Elias Leake Quinn. Smart metering and privacy: Existing laws and competing policies. *Available at SSRN 1462285*, 2009.

[66] G. Eibl, D. Engel, and C. Neureiter. Privacy-relevant smart metering use cases. In *2015 IEEE International Conference on Industrial Technology (ICIT)*, pages 1387–1392, 2015.

[67] Ulrich Greveler, Peter Glösekötterz, Benjamin Justusy, and Dennis Loehr. Multimedia content identification through smart meter power usage profiles. In *Proceedings of the International Conference on Information and Knowledge Engineering (IKE)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer . . . , 2012.

[68] S. S. Chawathe. Recognizing activities of daily living using binary sensors. In *2018 4th International Conference on Universal Village (UV)*, pages 1–6, 2018.

[69] Blase Ur, Jaeyeon Jung, and Stuart Schechter. The current state of access control for smart devices in homes. In *Workshop on Home Usable Privacy and Security (HUPS)*. HUPS 2014, July 2013.

[70] Mathieu Thiery, Vincent Roca, and Arnaud Legout. Privacy implications of switching on a light bulb in the iot world. 2019.

[71] Elad Salomons, Lina Sela, and Mashor Housh. Hedging for privacy in smart water meters. *Water Resources Research*, 56(9):e2020WR027917, 2020. e2020WR027917 2020WR027917.

[72] *Nuki - Door Sensor*, 2018 (accessed 20.05.2021). `https://nuki.io/en/support/smart-lock/mounting-on-the-door/doorsensor/`.

[73] S. Notra, M. Siddiqi, H. Habibi Gharakheili, V. Sivaraman, and R. Boreli. An experimental study of security and privacy risks with emerging household appliances. In *2014 IEEE Conference on Communications and Network Security*, pages 79–84, 2014.

[74] V. Sivaraman, H. H. Gharakheili, A. Vishwanath, R. Boreli, and O. Mehani. Network-level security and privacy control for smart-home iot devices. In *2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 163–167, 2015.

[75] Noura Abdi, Kopo M. Ramokapane, and Jose M. Such. More than smart speakers: Security and privacy perceptions of smart home personal assistants. In *Fifteenth Symposium on Usable Privacy and Security (SOUPS 2019)*, Santa Clara, CA, August 2019. USENIX Association.

[76] Ebenezer Hailemariam, Rhys Goldstein, Ramtin Attar, and Azam Khan. Real-time occupancy detection using decision trees with multiple sensor types. In *Proceedings of the 2011 Symposium on Simulation for Architecture and Urban Design*, SimAUD '11, page 141–148, San Diego, CA, USA, 2011. Society for Computer Simulation International.

[77] M. K. Masood, Yeng Chai Soh, and V. W. . Chang. Real-time occupancy estimation using environmental parameters. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2015.

[78] Johanna Richter, Andreas Bischof, Albrecht Kurze, Sören Totzauer, Mira Freiermuth, Michael Storz, Kevin Lefeuvre, and Arne Berger. Machtförmige praktiken durch sensordaten in wohnungen. *Mensch und Computer 2018-Tagungsband*, 2018.

[79] Eric Zeng, Shrirang Mare, and Franziska Roesner. End user security and privacy concerns with smart homes. In *Thirteenth Symposium on Usable Privacy and Security ({SOUPS} 2017)*, pages 65–80, 2017.

[80] R. Chow. The last mile for iot privacy. *IEEE Security Privacy*, 15(6):73–76, 2017.

[81] O. Alrawi, C. Lever, M. Antonakakis, and F. Monrose. Sok: Security evaluation of home-based iot deployments. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1362–1380, 2019.

[82] R. Perdisci, T. Papastergiou, O. Alrawi, and M. Antonakakis. Iotfinder: Efficient large-scale identification of iot devices via passive dns traffic analysis. In *2020 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 474–489, 2020.

[83] Python Software Foundation. *Python*, 2020 (accessed 20.05.2021). `https://www.python.org`.

[84] Timothy Libert. Exposing the hidden web: An analysis of third-party http requests on 1 million websites, 2015.

[85] Tim van Kasteren, Athanasios Noulas, Gwenn Englebienne, and Ben Kröse. *Accurate Activity Recognition in a Home Setting*, page 1–9. Association for Computing Machinery, New York, NY, USA, 2008.