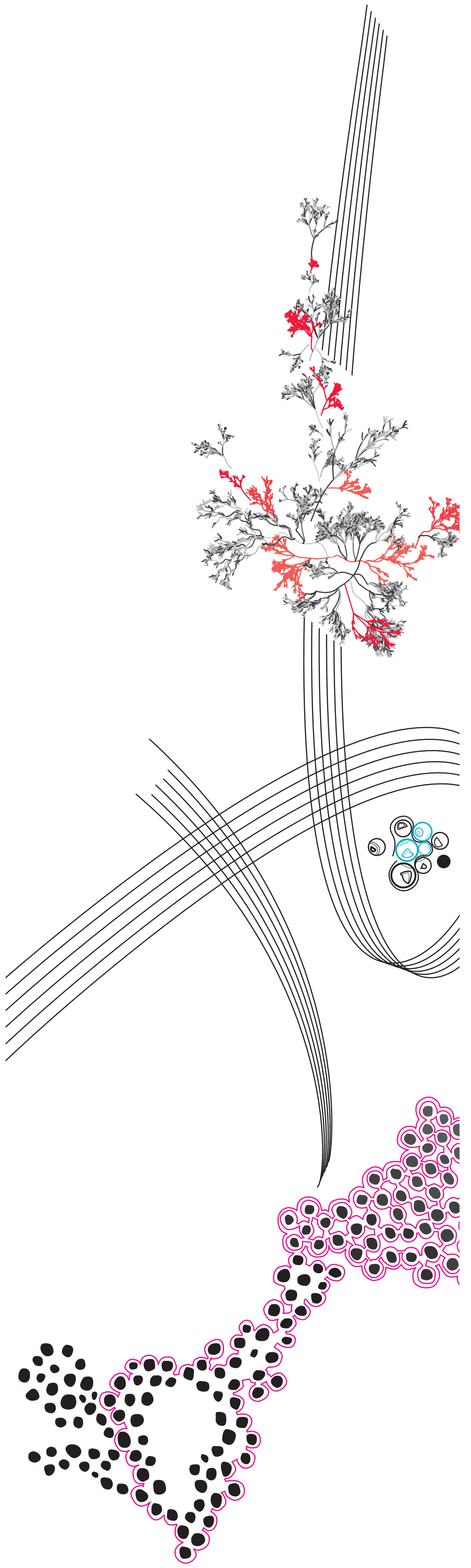
MSc Cybersecurity
Master Thesis



# Automated Generation of Security and Privacy Labels for IoT Devices: A Framework based on NIST Guidelines

Gabriele Abbate

Supervisor: Andrea Continella
Co-supervisor: Chakshu Gupta
Committee Member: Alex Chiumento

September, 2024

Department of Computer Science
Faculty of Electrical Engineering,
Mathematics and Computer Science,
University of Twente

UNIVERSITY OF TWENTE.

# Automated Generation of Security and Privacy Labels for IoT Devices: A Framework based on NIST Guidelines

Gabriele Abbate
University of Twente
P.O. Box 217, 7500 AE Enschede
The Netherlands
g.abbate@student.utwente.nl

***Abstract*—The proliferation of Internet of Things (IoT) devices has significantly transformed daily life by providing enhanced convenience and connectivity. However, this widespread adoption has also raised serious concerns about security and privacy. Despite the growing reliance on this technology, many regular users remain unaware of the risks associated with their IoT devices. In response, recent efforts, including guidelines from the National Institute of Standards and Technology (NIST), have aimed to improve security and privacy awareness through a proposed labeling system. While these guidelines offer a foundation, there remains a gap in effectively implementing and understanding security and privacy factors. Our study addresses this gap by proposing an automated methodology focused on network traffic analysis for extracting features from IoT devices to generate security and privacy labels. The results demonstrate that our methodology significantly improves the efficiency of feature extraction compared to manual methods, while also achieving high accuracy.**

***Index Terms*—Label, Internet of Things (IoT), Network Analysis, Privacy and Security.**

## I. Introduction

Internet of Things (IoT) devices have revolutionized various aspects of our lives by allowing for seamless connectivity and automation in everyday tasks, healthcare monitoring, and remote control of home appliances. However, they also come with many vulnerabilities that can be exploited. These vulnerabilities not only threaten the functionality of the devices themselves but also compromise user data, potentially exposing sensitive personal information. Consider the Verkada security camera hack [20] as an example. In 2021, the company suffered a major data breach, sparking serious privacy concerns. Cybercriminals gained access to over 150,000 live camera feeds and archived footage from various locations, including schools, hospitals, police stations, and companies like Tesla and Cloudflare. In contrast, other well-known attacks like the Mirai Botnet [18], Stuxnet [39], and the Jeep Cherokee Hack [17] primarily compromised users' security. In these cases, attackers took control of IoT devices to carry out large-scale cyberattacks. Such incidents underscore the critical need for robust security protocols to protect both the devices and the personal information they handle, thereby addressing growing concerns about the safety and privacy of IoT ecosystems.

In response to these disruptive events, consumers have grown awareness about the security and privacy implications of IoT devices. Consumers are increasingly concerned about how their data is collected, stored, and utilized [21].

Recognizing the importance of securing IoT devices, the European Union and the United States have taken steps to address these concerns [29, 34, 35]. The EU Parliament has highlighted the risks associated with IoT devices and emphasized the importance of implementing security-by-design approaches to ensure the security of connected products. The EU Cyber-Resilience Act [12], proposed by the European Commission, aims to regulate IoT products based on their level of risk. This Act requires the manufacturers to provide consumers with information about the security of their devices, in compliance with EU laws. Similarly, the EU and the U.S. have signed an Administrative Arrangement on a Joint CyberSafe Products Action Plan to cooperate more closely in developing common standards and labels for IoT devices [11]. As a result of the need to label IoT products to make users more aware of their devices' privacy and security, Emami-Naeini et al. [8, 10, 9] proposed a comprehensive list of factors to disclose, along with an accurate labeling system.

However, producing accurate privacy and security labels for IoT devices presents significant challenges. These labels often tend to be inaccurate due to factors such as developers' negligence, lack of knowledge about certain parts of the code, unexpected data collection from third-party libraries, and misinterpretations of terms in privacy labels [22, 24, 23]. These complexities make it difficult for vendors to self-assess the security and privacy of their products accurately. As a result, there is an increasing need for tools and methodologies that can automatically extract privacy and security factors from IoT devices, thereby reducing human error.

In response to these growing concerns, a few approaches have been explored to improve the accuracy of privacy and security label generation. In their work, Huang et al. introduce *IoT Inspector* [15], an open-source tool to observe the traffic of IoT devices. The tool can also automatically find some vulnerabilities such as weak TLS versions and weak ciphers through network analysis. On the other hand, even though it is not connected to the IoT ecosystem, Li et al. present *Matcha* [23], a tool used to assist developers in creating accurate Google Play data safety labels through automated code analysis. However, none of these approaches focus on any of the privacy and security factors proposed by Emami-Naeini et al. [8].

To address this need, we aim to fill this research gap by developing a methodology to automatically extract privacy and security-related factors from IoT devices required to generate security labels. These factors include data collection practices, communication protocols, encryption, and authentication mechanisms. Our study seeks to explore the feasibility and applicability of network analysis for automated extraction techniques.

In this research, we address the challenges of self-assessing privacy and security in IoT devices by introducing a novel methodology. As part of this approach, we present *SPIoT* (Security and Privacy in the Internet of Things), a tool designed to extract privacy and security factors from IoT devices using active and passive network analysis. *SPIoT* is an open-source tool developed in Python and is available on GitHub.

In summary, the contributions of this research over the state-of-the-art are:

- **Automated Extraction Methodology**: Development of a novel methodology to automatically extract privacy and security factors from IoT devices, reducing reliance on manual assessment and mitigating human error.
- **Design and Development of SPIoT**: The open-source tool *SPIoT* represents a practical solution for automated analysis, supporting the extraction of privacy and security factors. This tool will aid manufacturers in adhering to regulatory requirements such as the EU Cyber-Resilience Act by assisting them in discovering and representing the security and privacy characteristics of their devices.
- **Testing and Evaluation**: We evaluated the prototype in a laboratory to verify that the generated labels are correct. The tested devices included both internal devices from our lab and external devices from sources we found online.

## II. Background and Related Work

In 2017, the European Parliament highlighted the risks linked to IoT devices and the importance of securing them by implementing a security-by-design approach to every device [13]. Over the following years, the EU took multiple steps to improve the security of these devices, including a June 2021 resolution focused on improving cyberdefense cooperation and developing advanced cybersecurity research. These efforts culminated in the proposal of the EU Cyber-Resilience Act (CRA) by the European Commission in 2022, aiming to establish a robust security framework and harmonize cybersecurity requirements across member states. The act creates two product categories according to their level of risk: critical and non-critical. Critical products are further divided into two sub-categories, class I lower risk, and class II higher risk. All products belonging to this second class will be regulated via a third-party assessment. In contrast, non-critical products, representing 90% of digital products on the market, will only require a self-declaration of conformity [12]. This distinction is crucial for the purposes of this paper, as the designed methodology will assist manufacturers in generating the self-declarations of conformity. The act will make manufacturers responsible for ensuring that their products are digitally secure and will enable consumers to have greater information regarding the security of their devices. In 2024, the EU and the U.S. signed an Administrative Arrangement on a Joint CyberSafe Products Action Plan [11]. This agreement aims to promote closer cooperation in developing common standards and labels for IoT devices, providing consumers with clear and accessible information to make informed decisions about the products they use. This partnership not only aims to improve the already discussed cybersecurity practices worldwide but also reduces the burden on businesses to comply with different regulations.

Currently, vendors have no standardized method to self-declare the privacy and security issues related to their products. This leads to a series of problems that might hamper correct communication from the vendors, ending up affecting consumer awareness. One of the most significant issues is the difficulty vendors face in accurately identifying their devices' privacy and security factors. This challenge often arises from negligence, lack of knowledge about parts of the code they didn't write, unexpected data collection from third-party libraries, or misinterpretations of terms in privacy labels [1, 14, 22, 3, 24, 23]. This might lead to incorrect or incomplete representations of a device's security practices, potentially causing errors when translating this understanding into security labels.

The growing concern among consumers highlights the need for better transparency in IoT device privacy and security. Many consumers report being unable to find relevant information before purchasing an IoT device [10]. Research also shows that consumers are more inclined to buy devices when assured that their data will not be retained or shared with third parties [9]. They perceive a higher level of risk when labels indicate that data will be sold to third parties, while they feel more secure when labels clearly state that data will not be sold at all.

In response to these concerns, the NIST proposed a labeling system in the 2021 draft of the Baseline Security Criteria for Consumer IoT Devices, officially published in 2022 [29, 34]. The NIST outlines two guiding principles for an effective labeling approach in the second paper. First, the label should adhere to specific technical standards for IoT device security. These standards define the security features a device should have and ensure that the label accurately reflects these capabilities. Second, the label should be designed to be easily understood by a broad range of consumers. It should not require specialized cybersecurity knowledge, allowing consumers to make informed decisions about the security features of IoT devices. Regarding the label's structure, they propose a simple binary label to indicate that a product meets a baseline standard. Additionally, consumers should be able to access more details through online resources, such as a URL or a QR code, both before and after purchase.

To define the set of labels, Emami-Naeini et al. [8] conducted a series of interviews and surveys with privacy and security experts. The results show that an ideal labeling system should consist of two different layers as described in Table I.

| | Primary Layer |
|---|---|
| 1.1 | Privacy rating for the device from an independent privacy assessment organization |
| 1.2 | Security rating for the device from an independent security assessment organization |
| 1.3 | The date until which security updates will be provided |
| 1.4 | Type of data that is being collected |
| 1.5 | Type of sensor(s) on the device |
| 1.6 | Whether or not the device is getting cryptographically signed and critical automatic security updates |
| 1.7 | Types of physical actuators (e.g., talking, blinking) the device has and in what circumstances they are activated |
| 1.8 | Whether or not the device is using any default password |
| 1.9 | Frequency of data sharing (e.g., continuous, on demand) |
| 1.10 | The warranty period of the device |
| 1.11 | Level of detail (granularity) of the data being collected, used, and shared (e.g., identifiable, aggregate) |
| 1.12 | Access control for device and apps (e.g., none, single-user account, multi-user account) |
| 1.13 | Who the data is shared with |
| 1.14 | Who the data is sold to |
| 1.15 | Whether the data transits in plain-text, encoded, or encrypted |
| 1.16 | List of open ports |

| | Secondary Layer |
|---|---|
| 2.1 | Retention time |
| 2.2 | Purpose of data collection |
| 2.3 | What information can be inferred from the collected data |
| 2.4 | Supported standards (e.g., Wi-Fi, Zigbee) |
| 2.5 | Where the collected data is stored |
| 2.6 | Whether or not the collected data will be linked with data obtained from other sources |
| 2.7 | Special data handling practices for children's data |
| 2.8 | The control that users are offered (e.g., opt-in/out from data sharing) |
| 2.9 | Data-collection frequency (e.g., once a month, on install) |
| 2.10 | Whether or not the device can still function when Internet connectivity is turned off |
| 2.11 | Level of detail (granularity) of the data being collected, used, and shared |
| 2.12 | Relevant security and privacy laws and standards to which the device complies (e.g., ISO 27001, GDPR) |
| 2.13 | Link to the device's key management protocol |
| 2.14 | Resource usage in terms of power and data (e.g., kw, kbps) |
| 2.15 | The device manufacturer has a "bug bounty" program |
| 2.16 | Link to the software and hardware bill of materials |

TABLE I
PROPOSED DATA LAYERS REPRESENTING A COMPREHENSIVE LIST OF PRIVACY AND SECURITY FACTORS DIVIDED INTO PRIMARY AND SECONDARY LAYERS.

The same researchers also hosted a website, IoT Security & Privacy Label, that allows vendors to generate their own labels with the last design, based on the framework they proposed [7], under a Creative Commons CC0 license [6]. Many other websites that offer same services are found on the web, for example Privacy Policy Generator, App Privacy Policy Generator, and Free Privacy Policy Generator, which offer to businesses and individuals a labeling structure and allow a manual generation of privacy policies in compliance with CCPA, CPRA, GDPR, and with the requirements of Google Analytics & AdSense.

Despite the proposed solutions to improve transparency, the core issue of IoT vendors' inaccurate or incomplete privacy and security self-declarations persists. Even if the factors to extract from IoT devices have been identified, security and privacy self-declarations may still be inaccurate due to a lack of cybersecurity expertise among IoT developers. To address this issue, we need a methodology to automatically extract the privacy and security-related factors of devices, so that reliance on vendors' self-declarations is no longer necessary. The goal is to gather as much information as possible from active and passive network analyses to provide a comprehensive overview of all factors.

**Network Analysis**: The network analysis approach is the most accessible for automated information gathering. This is because network analysis can be universally applied to every IoT device, as every smart device needs to be connected to the internet. In contrast, other types of analysis, such as firmware and companion app analysis, have been considered as main approaches for feature extraction, but neither has proven suitable. Firmware analysis is limited by the diversity of firmware and the difficulty in developing a generic extraction methodology. Similarly, companion app analysis does not apply to devices that do not have an associated app.

As mentioned by Omar Alrawi et al. [2], despite several working groups proposing standardization for IoT devices, they have failed to reach a consensus on a solution. The underlying issue lies in the heterogeneity of home-based IoT devices. While core functionalities remain consistent, specific features such as hardware, firmware, or software can vary significantly across devices.

Due to the IoT ecosystem's complex structure, a full overview of how data is collected and shared is needed to perform a comprehensive network analysis. Specifically, the flow of sensitive data can occur between the device and the companion app running on a smartphone, between the device and the cloud, or between the app, which previously collected and processed the data from the device, and the cloud, as shown in Figure 1. Omar Alrawi et al. [2] proposed a methodology for the security evaluation of home-based IoT devices by intercepting traffic from the three connections mentioned above.

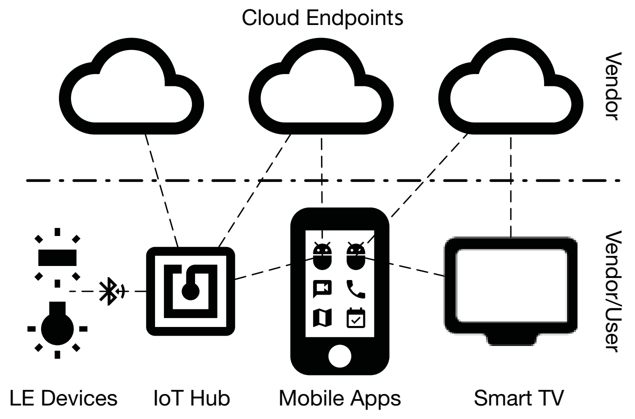


Fig. 1. Simple schema extracted from the paper by Alrawi et al. [2], illustrating how the flow of personal data can occur.

A similar security evaluation has been conducted by Tina Wu et al. [40], which assessed the same communication channels (device-to-cloud, mobile app-to-cloud, and mobile app-to-device). They listed the vulnerable open ports, assessed whether encryption had been used, and evaluated the robustness of the encryption protocol. Additionally, they performed an entropy analysis on the network traffic based on the work of Loi et al [25] to determine if the data in transit was in clear-text, encoded, or encrypted.

Regarding some influential tools for this category, *IoT Inspector* [15] deserves to be mentioned. It is an open-source software that allows users to identify potential security and privacy violations. Similar studies have been conducted on security and privacy analyses of IoT children's toys by analyzing network and application vulnerabilities [5, 38]. Other interesting research on IoT network analysis has been conducted by Mainuddin et al. [26] and M. Hammad Mazhar and Zubair Shafiq [28].

The reviewed literature reveals both progress and ongoing challenges in evaluating the security and privacy of IoT devices. Existing approaches have focused on assessing communication channels, encryption practices, and network traffic to identify vulnerabilities and privacy issues. Despite these advancements, there is a clear need for more consistent and automated evaluation methods to address the complexity of feature extraction. This indicates a crucial gap in current research and highlights the importance of developing a robust methodology for accurately assessing IoT devices' privacy and security features.

## III. Methodology

This work presents a methodology to automatically generate a security and privacy label for an IoT device by extracting information from network traffic. These labels serve as an informative summary of the device's privacy and security characteristics. In particular, they provide essential details such as the device name, the manufacturer's name, and a list of key factors crucial for understanding the device's privacy and security posture. The factors included in the label inform the user about various aspects, such as data encryption practices, third-party data-sharing policies, the presence of default credentials, and other critical security and privacy elements. The whole approach is inspired by and expands upon the work of Emami-Naeini et al. [8].

We extract the factors that can be identified through passive or active network analysis techniques, depending on the nature of the data being processed. Passive network analysis is performed by recording the device's network traffic without interference, which is then analyzed with the appropriate tools. Active network analysis is performed by probing the device to gather more specific information about its security features. The proposed methodology extracts the information from the traffic to populate the label by leveraging these network analysis methods. This approach provides consumers and regulators with transparent and reliable information about IoT devices' security and privacy practices.

Manufacturers can also benefit from this automated methodology, as it reduces human error and does not require extensive cybersecurity knowledge. Regular network traffic analysis cannot reveal what information is being shared with servers when data is encrypted. However, manufacturers have an advantage in this scenario, as they can access unencrypted traffic by testing their devices in debug mode and installing the necessary certificates. This allows them to use the automated tool to extract the required information for the labels without being limited by the encrypted traffic.

**Challenges**: The labels we aim to generate require detailed information about the devices' security and privacy practices. To obtain this information, we analyze the network traffic generated by the devices using passive and active network analysis techniques. While this approach allows us to extract a range of important factors, we encountered several challenges during the process.

One significant challenge is the presence of encrypted traffic, which makes it hard to determine the exact data being transmitted. This encryption poses problems not only for designing the methodology but also for evaluating its accuracy. Currently, we can extract only the unencrypted data. We would generate a fully comprehensive list of the data shared by the device if we could decrypt all the traffic.

Decryption would also let us distinguish between data that is shared in an encrypted form and data that is transmitted unencrypted. This distinction is crucial, as it directly impacts the level of privacy and security experienced by the users. By identifying which data is protected by encryption and which is not, we could better inform consumers about the privacy implications of using the device.

Another challenge arises from the lack of pre-existing privacy and security labels for IoT devices. As a result, we had to manually extract each feature for our study. This manual extraction process introduces some uncertainty, as it may be incomplete or imprecise due to the complexities of interpreting network traffic.

Our methodology focuses on extracting the following key factors: user data sent to the cloud, user data shared with third parties, data encryption, mutual authentication certificates, a list of open ports and services, and default credentials. However, it does not cover every factor listed in Table I, as the required information is not available through network traffic analysis. Some of these factors require a vendor's statement, while others need to be extracted through different types of analysis, such as firmware analysis or companion app analysis.

### A. Passive Analysis

*1) User Data Sent to the Cloud:* This factor focuses on the user-specific data the devices send to the cloud, which may or may not be sensitive. We categorize the sensitive data into four types: media, physiological, location, and other.

**Media Data**: The analysis of media data transmission focuses on identifying whether audio or video streaming is occurring from IoT devices to the cloud.

First, we examine the protocol used for the transmission. Certain protocols like RTP and RTSP are typically associated with audio streaming. We can infer that the data is audio if these protocols are detected. We need to investigate further if the primary protocol is UDP and no specific data type information is available. We analyze the proportion of UDP packets relative to the total number of packets; a high proportion suggests the presence of streaming data. We also perform an entropy analysis to confirm this. Entropy analysis is a measure of randomness in data packets. Continuous data streams, such as audio or video, are typically high-entropy due to their variable sequences and the wide range of possible values. This complexity arises because complex information, such as sound waves or image pixels, needs to be encoded; this automatically yields a more stochastic distribution of data [36].

Conversely, low entropy tells us that the data reflects more structure or predictability, often implying the transmission of simpler kinds of data or non-continuous data. By carrying out the entropy analysis, we add to our hypothesis that a high ratio of UDP packets is likely to be streaming data. High entropy in those packets would increase the likelihood that the data sent is complex, involving formats like audio or video streams, rather than other data types that are less complex and have more structure. Therefore, entropy analysis gives us another layer of confirmation for accurately pinpointing the type of data transferred.

Finally, To differentiate between audio and video streams, we calculate the average size of data transmitted over a defined period of time. Video streams typically transmit larger amounts of data compared to audio streams, due to the greater complexity and information density of video content.

The video data are formed from a continuous sequence of frames. Each frame includes information on all the pixels, and even after compression, these require vastly more bandwidth and storage than the audio signals. Audio, on the other hand, is usually much smaller in size because sound waves can be represented with less data. Techniques for compressing audio are also more effective, thus continuing to decrease the overall size of audio streams compared to that of video.

We can use this property to classify the media data by applying a size threshold. If the cumulated size of streamed data exceeds a threshold value, the content is likely video, while smaller data sizes tend to indicate audio streaming. This allows distinguishing between these two classes of media in cases when protocol analysis is insufficient.

**Physiological Data**: The extraction of physiological data involves identifying and categorizing various personal and health-related metrics transmitted to the vendor's cloud. Devices that typically collect this type of data include wearable fitness trackers, smartwatches, and health monitoring devices. The data includes personal information (height, weight, age, gender), vital signs (heart rate, blood pressure, respiratory rate, body temperature), activity levels (steps taken, calories burned, exercise intensity), medical data (blood glucose levels, oxygen saturation), and wellness metrics (stress levels, mood indicators, fatigue levels). Data extraction is performed by analyzing the primary protocols used for data exchange. Manual inspection of network captures indicated that the most frequently used protocol is HTTP. Other common protocols used by IoT devices for data exchange include MQTT and Telnet. These protocols have a standardized structure and the data can be statically extracted. For example, MQTT organizes information in a topic (the type of data) and a message (the actual data content). Figure 2 shows a standard MQTT packet format.

```
Msg Len: 2
Topic Length: 12
Topic: Stress Level
Message: 42
```

Fig. 2. Standard MQTT payload.

Similarly, Telnet sends commands and responses in a standard format, allowing data extraction by looking at the keyword "Data" in the payload. Thus, extracting data from Telnet is straightforward since no additional parsing is needed.

When dealing with HTTP traffic, which does not follow a uniform structure, the data payloads are sent to a Large Language Model (LLM) for analysis. The LLM is used to parse and extract sensitive information from these varied formats, such as JSON or XML, which are not easily handled by static extraction methods.

**Location Data**: Location data primarily consists of GPS coordinates, including latitude and longitude. The data extraction and classification process is the same as described above.

**Other Data**: A fourth section has been created for the sensitive data that is not classified in any of the above categories. The "Other Data" category includes a wide range of information, including network data (IP addresses, MAC addresses, network traffic data, connectivity logs), sensor data (temperature, humidity, light levels, air quality, pressure, battery level), configuration data (device settings, customization options, user preferences, firmware version), and authentication data (username, password). The extraction methodology for this data also leverages standard protocols such as MQTT and Telnet, and the more generic HTTP. Also in this case the data is extracted from the payload and then categorized under "Other Data."

*2) User Data Shared with Third Parties:* In addition to the data being sent to the vendor's cloud, there is a critical concern regarding user data being shared with third parties. What differentiates this second factor from the first is that the destination address of the transmitted data is not associated with the vendor's domain but with third-party organizations. The list of vendor organizations must be included in the configuration file to extract this factor.

This classification is performed by resolving the IP addresses of the data recipients[1] and checking them against the configuration file to determine if they belong to external entities rather than the vendor. The resolved IP address provides information about the server's hostname, organization name, city, region, and country. If the organization is not listed as one of the vendor's organizations, the data is flagged as potentially being shared with a third party. The data extraction and classification process follows the same steps as described in the previous section.

*3) Data Encryption:* The analysis of data encryption focuses on determining whether the user's data transmitted by IoT devices is protected through encryption. If encryption

[1]https://ipinfo.io/

is applied, we determine whether such encryption is applied partially to some data transmission parts or all, and we verify if the applied encryption uses strong standards, such as TLS 1.2/1.3. Since encryption is a crucial security measure in protecting data from unauthorized access during transmission, its understanding is vital in assessing the general security of IoT devices. In this case, our methodology includes an assessment of data encryption by checking PCAP files for the presence of encryption protocols and unencrypted communications.

*4) Certificates for Mutual Authentication:* Evaluating mutual authentication in IoT devices involves checking whether the device uses cryptographic certificates issued by a Certificate Authority (CA). Mutual authentication ensures that not only does the server present a certificate to be authenticated by the client, but the client also shows its certificate during the TLS Four-Way handshake to be authenticated by the server. This bidirectional authentication improves security by confirming the identities of both communicating parties [16]. We analyze the PCAP files to search for certificates exchanged during the handshake to determine if the device is cryptographically signed and supports mutual authentication.
**Certificate Search**: During the TLS/SSL handshake, certificates are exchanged between the client and server. We can identify whether mutual authentication is implemented by inspecting the PCAP files for these certificates. The presence of certificates in the PCAP indicates that the device is cryptographically signed.
**Analysis and Classification**: If certificates from both the client and server are found in the PCAP, it confirms that the IoT device supports mutual authentication. If only the server's certificate is found, it indicates that mutual authentication is not implemented, as the client does not present its certificate for authentication by the server.

### B. Active Analysis

*1) List of Open Ports and Services:* This factor provides a list of all network ports and listening services, pointing out how the device can be accessed. The result of a network scan shows the open ports on the device and the related listening services, along with their version.

*2) Default Credentials:* After determining how the user can access the device, this phase focuses on identifying the use of default credentials. Many IoT devices, such as routers, come with default passwords that are insecure if not changed by the customer. Other IoT devices may also be vulnerable to unauthorized access through weak credentials, as demonstrated by Kumar et al. [19], by performing dictionary attacks against FTP and Telnet services.

To check if the IoT devices under analysis accept default credentials, we conduct dictionary-based attacks against the identified services. If any default credentials provide access, the device is labeled "*The device can be accessed with default credentials*." Otherwise, it is labeled "*The device cannot be accessed with default credentials*."

## IV. Implementation

*SPIoT* is structured with several distinct modules, each designed to extract specific security and privacy factors. The tool operates based on a configuration file created for each device, which contains crucial information for its functionality. This configuration file includes the path to the PCAP, the device's MAC address, its name, the IP resolver API token, the LLM API token, and vendor details.

The section about vendor details is essential, as it encompasses the vendor's name and a list of organizations associated with the device's cloud service. This information helps determine whether interactions with the device are handled by the proprietary cloud provider or a third-party entity.

The architecture of *SPIoT* is divided into several modules, each responsible for a specific aspect of feature extraction. The modular design ensures that *SPIoT* can extract each factor independently, without relying on module dependencies. This architecture also allows IoT developers to adapt the tool to their needs and select the factors they want to include.

### A. Passive Analysis

The first phase of feature extraction analyzes network traffic in PCAP format. Each feature in this section is extracted by filtering the relevant traffic with tshark, based on the source MAC address and the protocols needed by each specific module. The decision to use tshark instead of Pyshark was made after comparing their performances and finding that tshark was significantly faster than the Python library.

*1) User Data Sent to the Cloud:* The configuration file is loaded, and the analysis begins by examining the HTTP packets sent by the IoT device. The traffic is filtered and divided into HTTP streams using tshark. Managing the entire data stream instead of just individual payloads facilitates data extraction for the LLM since more context is provided. Once extracted from the data stream, the payload is converted from hexadecimal data to bytes and decoded to a UTF-8 string. The readable data is passed to the LLM assistant, which extracts the sensitive data by analyzing the semantics of the stream. The extracted data is then saved in the format "data_type:data_value" (e.g., "calories burned: 340"). This data is classified into the appropriate category (e.g., physiological data) by passing it again through the LLM, which acts as a classifier in this instance.

The LLM used for this process is LLaMa3.1 [30] by Meta, specifically the 8B and 70B parameter versions. We used the Ollama [37] Python library during the implementation phase to run LLaMa3.1-8B. We used the OpenAI client with LlamaAPI [31] to run LLaMa3.1-70B for the final performance evaluation.

The data extraction prompt for the LLM includes specific instructions on the exact type of data to return. The list of sensitive data to be identified is outlined in Table II. We arbitrarily selected the items to include in the list, as determining which data is considered "sensitive" is beyond the scope of our research. A real-world scenario would require manufacturers to follow official guidelines and adjust the LLM prompt accordingly.

The extracted data is returned in JSON format, following the structure shown in Figure 3.

Every data is then stored under the appropriate IP destination address to track where it is being sent.

```
{
  "Type of data 1": "Content of data 1",
  "Type of data 2": "Content of data 2",
  ...
  "Type of data N": "Content of data N"
}
```

Fig. 3. Example of the JSON format used for data extraction.

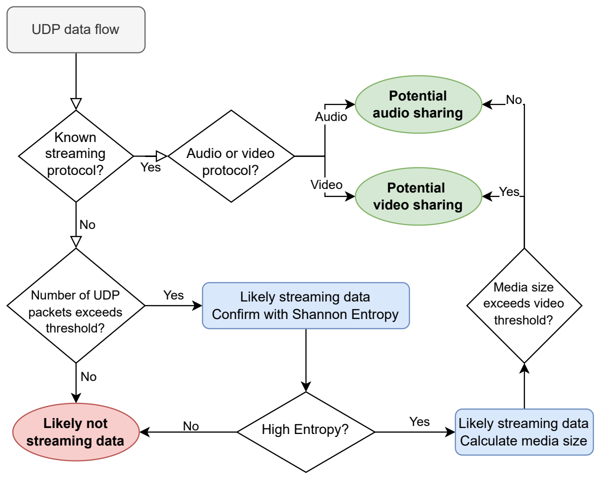
Regarding streaming data, the identification process leverages protocol information and packet analysis to infer the type of media data being transmitted. Firstly, if the highest layer protocol used in the transmission is specified, the data type can be inferred directly. For instance, protocols like Real-time Transport Protocol (RTP) and Real-Time Streaming Protocol (RTSP) indicate audio data transmission. If one of these protocols is detected, the function reports *"Potential audio or video sharing detected via x protocol"* and stores the result in a JSON file, including the destination IP to identify the receiver.

However, in case the highest layer protocol is simply User Datagram Protocol (UDP), a more detailed analysis of the packets is required. The approach here involves calculating the ratio of the total number of packets to the number of UDP packets. A high ratio indicates that most traffic is carried over UDP, suggesting potential media streaming activity. An entropy analysis is performed on the network traffic to further confirm the nature of the streaming data. As explained previously, high entropy levels indicate that the data stream contains a complex, non-repetitive pattern, characteristic of streaming media data. This step ensures that recurring patterns, such as those found in encoded or compressed data, do not mislead the identification process. Lastly, to differentiate between audio and video streaming, we consider the average size of data transmitted over a time interval. Using a threshold $t$, if the average data size exceeds the threshold, the stream is classified as video; if it falls below the threshold, it is classified as audio. Figure 4 describes the streaming detection process.

Some other protocols extensively used by IoT devices, such as MQTT and Telnet, can be easily handled thanks to their standardized formatting. We implemented an ad-hoc function for each of these protocols to extract and save the shared data into a JSON file, including the destination IP address of the recipient. In this case, if any of these protocols is found through research with tshark, the corresponding function is called and the data is extracted according to the protocol used.

Lastly, each data needs to be classified into one of the four categories: media, physiological, location, and other. For this task, we initialize a dictionary with the four predefined categories. Each category contains information about whether data is stored in the cloud or shared with third parties as shown in Fig 5.

The function iterates over the data dictionary, classifying each data type using the LLM assistant, which acts as a data classifier this time rather than a data extractor. Depending on the classification and the destination of the data, it updates the dictionary to reflect whether the data is stored in the cloud or shared with a third party.



UDP data flow
Known streaming protocol?
Yes
Audio or video protocol?
Audio
Video
Potential audio sharing
Potential video sharing
No
Yes
No
Number of UDP packets exceeds threshold?
Yes
Likely streaming data Confirm with Shannon Entropy
Media size exceeds video threshold?
No
Likely not streaming data
No
High Entropy?
Yes
Likely streaming data Calculate media size

Fig. 4. This flowchart describes the process of streaming data identification in UDP data flows. It begins with checking for known streaming protocols. The media type (audio or video) is determined if a known protocol is detected, and potential sharing is flagged. The number of UDP packets is analyzed against a threshold if no known protocol is detected. High UDP packet volume leads to entropy analysis to confirm streaming data. If high entropy is detected, the media size is calculated to distinguish between audio and video streaming.

```
{
   "Media": {
      "Stored in the cloud": "No media data is
           stored in the cloud",
      "Shared": "No media data is shared with
           any third party"
   },
   "Physiological": {
      "Stored in the cloud": "No physiological
           data is stored in the cloud",
      "Shared": "No physiological data is
           shared with any third party"
   },
   "Location": {
      "Stored in the cloud": "No location data
           is stored in the cloud",
      "Shared": "No location data is shared
           with any third party"
   },
   "Other": {
      "Stored in the cloud": "No other data is
           stored in the cloud",
      "Shared": "No other data is shared with
           any third party"
   }
}
```

Fig. 5. Example of the JSON format used for data classification.

*2) User Data Shared with Third Parties:* The extraction of this feature builds upon the analysis methodology used for data sent to the cloud but adds a focus on the data destination.

After extracting the data and categorizing it as described in the previous section, the next critical step is to resolve the IP addresses of the data recipients. Using the ipinfo.io API, every destination IP address is translated into its geographical and organizational information. The resolved information is then compared with the list of proprietary vendor domains from the configuration file. Data is classified as potentially shared with a third party if the IP address belongs to an external organization that is not listed as part of the vendor's infrastructure.

The overall classification structure is expanded in this section to separate the data shared with third parties from the data collected by the vendor's cloud. The dictionary used for categorization (see Figure 5) is updated with fields that indicate whether the data has been shared with external entities.

*3) Data Encryption:* To determine the amount of encrypted traffic from an analyzed IoT device, packets are categorized by their protocols. This analysis uses tshark, which filters and counts packets based on these protocols. One function identifies encrypted traffic by filtering packets associated with protocols like TLS, SSL, QUIC, ESP, AH, DTLS, and SSH. Another function identifies unencrypted traffic by filtering packets with the same MAC address associated with protocols such as HTTP, FTP, MQTT, and Telnet.

The results from the two functions are compared to determine the encryption status of the traffic:

- If only encrypted packets are found, the label "*All traffic is encrypted*" is generated, indicating that all data transmitted is protected through encryption.
- If both packets containing encrypted and unencrypted protocols are detected, the result is labeled as "*Traffic is partially encrypted.*" This indicates that some portion of the data is encrypted while other parts are transmitted in clear text.
- If only unencrypted packets are detected, the result is labeled as "*All traffic is unencrypted*", showing that no encryption is applied to the transmitted data.

The output of this analysis provides information about the encryption status based on the entire traffic capture. It does not offer granularity at the flow level, as the primary goal of this feature is to conduct a high-level assessment of overall network encryption rather than a detailed traffic analysis.

*4) Certificates for Mutual Authentication:* The implemented function extracts and analyzes SSL certificates linked to the IoT device's MAC address. It uses tshark to filter packets containing certificate messages (SSL handshake type 11) and the device's MAC address. If at least one certificate is found, the device is labeled as "*cryptographically signed.*" If no certificates are found, the device is labeled as "*not cryptographically signed.*"

### *B. Active Analysis*

We employ active network analysis on IoT devices to extract the last factors this research aims to find. Unlike passive analysis, which relies on capturing and inspecting network traffic, active analysis involves direct interaction with the device. This approach allows us to identify open ports and services and verify potential security weaknesses such as accepting default credentials.

*1) Open Ports and Services:* The scan uses *Nmap* to check TCP and UDP ports, ensuring all services are discovered. The output is processed to identify open services by splitting the Nmap output into lines and filtering for those containing the string "*open.*" The port number and service name are extracted from each of these lines, and then organized into a list of lists, with each sub-list containing a port number and its corresponding service name. The final label displays the device's list of open ports and services.

*2) Default Credentials:* This function performs dictionary-based attacks on various network services to identify default credentials. It starts by defining the IP address of the target device and the file paths for credential lists specific to FTP, SSH, and Telnet services. The function uses *Hydra* to attempt logging in to these services with a list of credential pairs. The lists of default credentials can be found on GitHub in danielmiessler's SecLists repository [32].

Using the list of open ports and services generated previously, the attack is executed on the appropriate services: SSH, FTP, Telnet, and HTTP(S). Hydra is launched with the corresponding credential list for SSH, FTP, and Telnet, and then the output is printed. For HTTP and HTTPS services on ports 80 and 443, *Nmap* with specific scripts (http-form-brute for HTTP and http-brute for HTTPS) is used to perform the brute-force attack. The results from these Nmap commands are captured and printed. The lists of default passwords, found on GitHub [33], are tailored for each service to improve accuracy and reduce waiting times by providing an ad-hoc solution for each service.

## V. Dataset and Experimental Settings

The UT IoT lab is configured to emulate a studio apartment with 23 IoT devices. Traffic capture is configured at the router level, ensuring that all TCP/IP traffic from the devices is captured as they communicate through the router. The data collection process was conducted over a continuous 48-hour period, resulting in a single data flow that was saved as a single PCAP file containing approximately 2 billion packets. To simulate realistic usage patterns, we interacted with each device as a typical user would, performing basic interactions such as turning devices on and off, adjusting settings, and triggering their functionalities, replicating regular household use with multiple users.

The experiments were conducted on all the devices installed in the lab, but the manual testing to verify the results was performed only on the Amazon Echo Dot and the Ring Doorbell. These two devices were specifically chosen for manual testing because they are the only ones that transmit partially encrypted traffic. In contrast, the other devices share their data entirely in encrypted formats.

Due to the large amount of encrypted traffic found in the UT Lab devices, we utilized a publicly available dataset to test the performance of the LLM assistant. The dataset, called IoT Devices Captures [27], was created by Aalto University and contains the traffic emitted during the setup of 31 smart home IoT devices. This dataset, which includes older IoT device traffic, provides some unencrypted data flows. The lack of encryption allowed us to effectively test our methodology in a scenario where traffic can be fully analyzed.

We specifically chose the PCAPs from two devices in the dataset to evaluate the LLM assistant. The first was captured during interaction with D-Link's HD IP Camera DCH-935L, and the second with the Wireless Scale WS-30 by Withings. These two PCAPs were selected because, after manually inspecting each file in the dataset, we found that they contained the highest amount of information shared in clear text.

## VI. Experimental Evaluation

The accuracy and efficiency of *SPIoT* is evaluated by comparing its results with those obtained through manual extraction. The same traffic files are used for both manual and automatic analysis. The manual analysis involves inspecting the PCAP files with Wireshark to extract the factors without the aid of automated tools. This process includes reading the packets sent by the IoT device, examining the payloads, headers, protocols, and traffic flow.

Decrypting TLS traffic to read the content of encrypted packets is impossible in our scenario. The encryption keys required for decryption are stored securely on the IoT devices and servers, making them inaccessible for external inspection. Without these keys, the content of encrypted transmissions cannot be deciphered.

Given the impossibility of decrypting the TLS traffic, the devices used for testing the collected and shared data were chosen based on the amount of unencrypted data sent to the server. In a real-case scenario, the vendor would be able to run the label generation in debug mode by installing the required certificates, which would allow the decryption of the entire network capture.

The testing methodology involves comparing, for each feature, the information retrieved manually with that extracted automatically. The final objective is to assess how well the automated extraction methodology replicates the outcomes of the manual analysis. If the two methods produce very similar results, it indicates that the automated system is accurate and reliable for extracting the required data.

### A. Passive Analysis

*1) Shared and Collected Data:* The first device analyzed is the *Amazon Echo Dot*. The manual analysis began by inspecting the data. While all HTTP traffic is protected by TLS v1.2, the high volume of UDP packets the device sends suggests media streaming. Online research confirmed that the device collects some media files and that the media is not video, as the device lacks a camera. It was further determined that the media being streamed to the cloud is audio captured through the embedded microphone. Given that all traffic is encrypted except for the UDP packets, we can conclude that the traffic is only partially encrypted and that the vendor collects some audio. In particular, hundreds of short audio streams, each lasting a few seconds, were sent to the cloud over 24 hours of recording. We separated each stream based on the time elapsed between them, considering streams separated by more than one second as non-continuous and thus distinct. The automatic data extraction successfully identified the audio streams, detecting 11,358 audio streams and zero video streams. It classified the data as *"Stored in the cloud"* after resolving the destination address and verifying it against the list of cloud servers manually added by the manufacturer in the configuration file.

The second device we analyzed is the Ring Doorbell. As with the previous device, all traffic seems to be encrypted by TLS v1.2 except for the UDP packets. The hypothesis that these UDP packets were used for video streaming was confirmed by consulting the Ring website's FAQ[2], which states that Ring videos are stored in the cloud for up to 180 days. We identified 16 potential video streams by manually inspecting the network traffic. Once again, *SPIoT* detected the video streams sent to the vendor's cloud. Specifically, the output showed that the tool found 16 video streams and zero audio streams.

As mentioned, an external dataset was used to test the LLM assistant. The first PCAP is generated by the *D-Link HD IP Camera DCH-935L*. The traffic sent by this device is partially encrypted, but we found some sensitive data being shared in plain text. The adopted protocol for sharing the data is HTTP/XML. All the data we gathered by manually inspecting the packets in Wireshark is reported in Table III.

From this data, and according to the list of sensitive data reported in Table II, we identified the following items to be sensitive, all of them being part of the category "Other Data":

- Device Name
- Vendor Name
- Model Name
- Firmware Version
- MAC Address

To evaluate its accuracy, we tested the methodology using two versions of the LLaMa3.1 LLM, one with 8B parameters and the other with 70B parameters. The sensitive data we extracted manually represents the ground truth with which we compared the automatically generated results. We ran each model 10 times and compared the results with the manually extracted data. We assessed the performances according to the following metrics:

- **True Positives (TP)**: The LLM correctly identified data as sensitive, accurately matching the ground truth and its category.
- **False Positives (FP)**: The LLM incorrectly identified non-sensitive data as sensitive, which does not match the ground truth.
- **True Negatives (TN)**: The LLM correctly identified data as non-sensitive, which aligns with the ground truth of not being on the sensitive data list.
- **False Negatives (FN)**: The LLM failed to identify data that should have been classified as sensitive according to the ground truth, or confused its category.

The automatic analysis results varied significantly depending on the number of parameters used by the LLM. Specifically, LLaMa3.1-8B had difficulty accurately extracting and

[2]https://ring.com

categorizing sensitive data into the appropriate categories. Out of 10 runs, it experienced hallucinations in 5 instances, incorrectly saving data not present in the payloads. Additionally, it had difficulty distinguishing between sensitive and non-sensitive data and classifying it correctly. The average accuracy for LLaMa3.1-8B, when analyzing the D-Link HD IP Camera DCH-935L, is 0.7963.

In contrast, LLaMa3.1-70B demonstrated much higher accuracy in data extraction and categorization. Across all 10 runs, LLaMa3.1-70B showed no hallucinations and consistently categorized the data correctly under the "Other Data" category. Although it generally identified all sensitive data, it missed one sensitive information in 2 out of the 10 runs. These results show significantly greater reliability and precision compared to its 8B counterpart, with an average accuracy of 0.9941. The module took an average of 23.95 seconds, including data extraction, classification, and destination IP resolution. The plot in Figure 6 shows the accuracy comparison across the 10 runs.

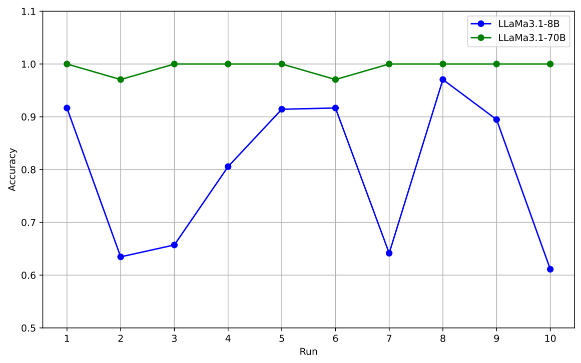


Fig. 6. Accuracy comparison of LLaMa3.1-8B and LLaMa3.1-70B across 10 runs. The plot illustrates the accuracy of both models in extracting sensitive data from unencrypted payloads sent by the D-Link HD IP Camera DCH-935L. It visually demonstrates that LLaMa3.1-70B consistently outperforms LLaMa3.1-8B in terms of accuracy, achieving a score of 0.9941 compared to 0.7963 for the 8B model.

To further confirm the better performance of the 70B parameter model, Figure 7 shows both models' average precision and recall.

The second PCAP from the IoT Devices Captures dataset we analyzed to test the LLM assistant is generated by *Withings Wireless Scale WS-30*. The PCAP analysis shows that the data is shared via HTTP through POST requests. Unlike the D-Link Camera, the adopted protocol is not HTTP/XML. Instead, the payloads are HTTP/URL encoded and appear as shown in Figure 8.

Among all the shared data, we identified the following sensitive information that was shared with the vendor's cloud:

- MAC Address: "00:24:e4:24:80:2a",
- Current Firmware: "881",
- Battery Level: "95",

As before, LLaMa3.1-8B had issues correctly identifying the sensitive information and understanding what data was being shared in the payloads. The 8B model experienced several

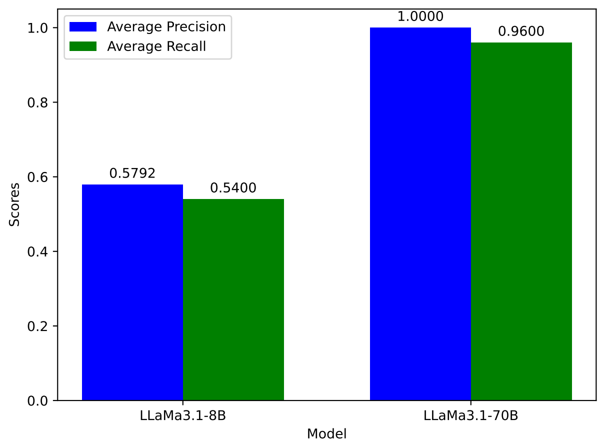


Fig. 7. Average precision and recall comparison for LLaMa3.1-8B and LLaMa3.1-70B. LLaMa3.1-8B exhibits low performance, with an average precision of 0.5792 and an average recall of 0.54. In contrast, LLaMa3.1-70B achieves much higher scores, with an average precision of 1.0 and an average recall of 0.96.

```
action=new&auth=00:24:e4:24:80:2a&hash=a05
d4a77ee5d4e0a02bc01f611450546&mfgid=294928
&currentfw=881&batterylvl=84&duration=300&
zreboot=1&trigger=weather&enrich=t
```

Fig. 8. Withings Wireless Scale WS-30 HTTP/URL encoded payload.

hallucinations and sometimes failed to recognize what data was being transmitted by the device. The average accuracy for this model is only 0.3630.

One major factor contributing to the smaller model's poor performance is the payload format. In the previous test, the data types were much clearer and more structured, making it easier for the model to identify and classify the sensitive information. However, in this second scenario, the data shared by the device is far more ambiguous, presenting a significant challenge. The lack of clarity in the payload made it difficult for the LLaMa3.1-8B model to extract the relevant information accurately, resulting in many misclassifications.

Conversely, LLaMa3.1-70B successfully extracted the three pieces of sensitive data 8 times out of 10 and consistently classified them as "Other Data." Additionally, the 70B model demonstrated an ability to enhance the readability of the data by renaming data types to more understandable names. For example, it reported "currentfw" as "Current Firmware" and "batterylvl" as "Battery Level." Also this time, the 70B model did not experience any hallucinations during the analysis. The average accuracy for LLaMa3.1-70B was 0.9875, confirming its superior capacity to handle more complex and unclear data payloads. Regarding the elapsed time, the execution took 23.38 seconds on average to extract all data from the payloads, filter the sensitive data, classify it into the correct category, and resolve the destination IPs. The plots in Figures 9 and 10 compare the two models' accuracy, precision, and recall.

*2) Data Encryption:* The manual inspection of the PCAP generated by capturing the traffic of the Amazon Echo Dot showed that, besides the UPD packets sent for the audio

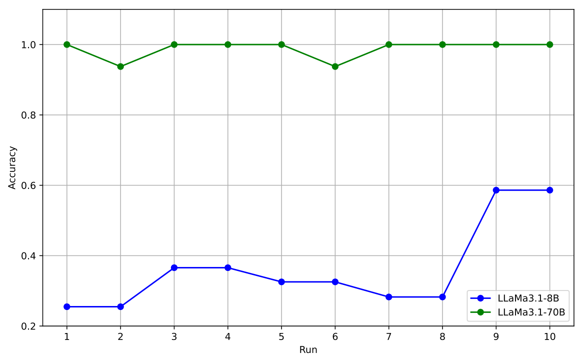


Fig. 9. Accuracy comparison of LLaMa3.1-8B and LLaMa3.1-70B across 10 runs. The plot illustrates the accuracy of both models in extracting sensitive data from HTTP/URL encoded payloads sent by the Withings Wireless Scale WS-30. Also this time LLaMa3.1-70B achieves better results than LLaMa3.1-8B in terms of accuracy, achieving a score of 0.9875 compared to 0.3630 for the smaller model.

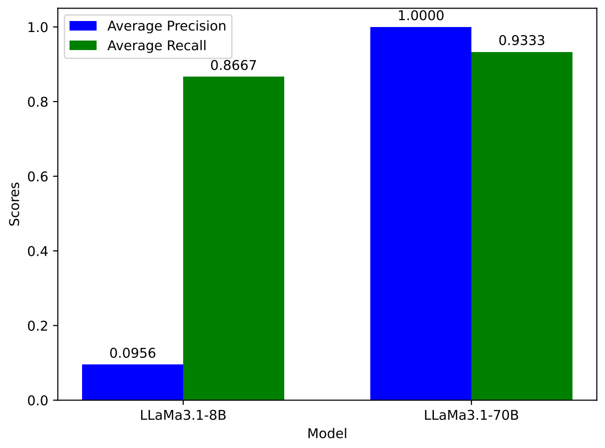


Fig. 10. Average precision and recall comparison for LLaMa3.1-8B and LLaMa3.1-70B in extracting sensitive data from unencrypted payloads sent by the Withings Wireless Scale WS-30. LLaMa3.1-70B achieves an average precision of 1.0 and an average recall of 0.9333, showing its ability to identify sensitive data with minimal errors. On the other hand, LLaMa3.1-8B shows a strong contrast in performance, with an average precision of only 0.0956, despite its relatively high recall of 0.8667. This indicates that although the smaller model can often detect sensitive data, it frequently misclassifies other data as sensitive, leading to a significant drop in precision.

streaming, all the traffic is encrypted. The tool successfully detected this and reported that *"Traffic is partially encrypted."* This classification reflects the tool's overall assessment of the entire traffic capture, rather than an analysis of individual flows. In this case, the tool identified encrypted and unencrypted UDP traffic were present in the same traffic capture. Additionally, It correctly reported that the negotiated TLS version is v1.2 and that the negotiated cipher suites are TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 and TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384.

Like the first device, the Ring Doorbell shares partially encrypted traffic. The video is streamed on an unencrypted channel, while all other data is kept confidential through TLS. The tool detected that the traffic is partially encrypted, that the negotiated TLS version is v1.2, that the negotiated cipher suites are TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 and TLS_RSA_WITH_AES_128_GCM_SHA256, of which the second is considered weak, according to the list of weak cipher suites published by Belshe et al. [4].

To better assess the accuracy of the encryption detection module, we tested it on every device in our IoT lab. As mentioned in Section V, the traffic capture was generated by 23 IoT devices and saved in a single PCAP file. The evaluation revealed that the tool correctly identified the encryption status of all 23 devices, with an average processing time of 0.67 seconds per device. Among them, only 3 devices produced partially encrypted traffic, while the remaining 20 generated fully encrypted traffic. Given the correct identification across all devices, the tool achieved 100% accuracy in detecting the amount of encryption. However, it is important to note that a larger dataset could provide more robust validation of the tool's performance, and further evaluation needs to be done.

*3) Certificates for Mutual Authentication:* The final factor to be extracted through passive analysis is the presence of a valid certificate issued by a Certificate Authority (CA). This certificate allows mutual authentication, ensuring that both parties involved in the communication can verify each other's identity.

When analyzing the traffic of the Amazon Echo Dot, we observed that the device supports mutual authentication. During the TLS handshake, another Amazon device, the Amazon Echo (192.168.11.49), requests the client's certificate, and the Amazon Echo Dot (192.168.12.43) responds by sending its certificate, as shown in Figure 14 in the Appendix. The automatic analysis successfully detected that the device holds a certificate for mutual authentication and accurately generated the corresponding feature.

Regarding the Ring Doorbell, no certificate was found during the TLS handshake, meaning the device's identity was not verified by either the server or other devices. *SPIoT* also did not detect the presence of any certificate on the device, so the generated feature correctly indicates that the device does not possess a certificate.

To further evaluate the accuracy of the methodology in detecting certificates for mutual authentication, we analyzed every device in the lab and compared the results of the automatic analysis with those from the manual analysis. Table V shows the full list of devices and their respective mTLS support status. The manual analysis revealed that, out of the 23 IoT devices tested, only the Amazon Echo Dot and the Philips TV supported mutual authentication, while the remaining 21 devices did not possess this feature. The automatic analysis accurately replicated this result with an average time of 0.56 seconds per device. The tool correctly identified the single cryptographically signed device and confirmed the absence of certificates for the other 21 devices.

In terms of performance metrics, the automatic analysis achieved 100% accuracy in this test setup. However, as we mentioned in the previous evaluation, testing this module on a larger set of devices would be ideal to obtain a more realistic and generalizable accuracy score. The confusion matrix is shown in Figure 11.

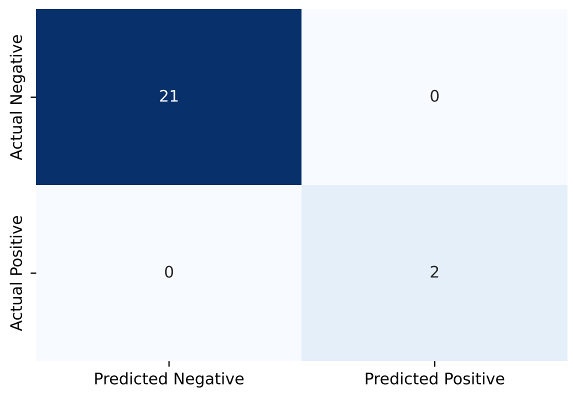


Fig. 11. Confusion matrix illustrating the results of the mTLS detection methodology. The matrix compares the outcomes of the automatic analysis against the ground truth determined by manual inspection. The analysis achieved 100% accuracy in detecting mutual TLS support across the 23 IoT devices, correctly identifying the two devices with mTLS certificates (Amazon Echo Dot and Philips TV) and the 21 devices without.

## B. Active Analysis

*1) Open Ports and Services:* Active network analysis to check open ports and services is performed using Nmap.

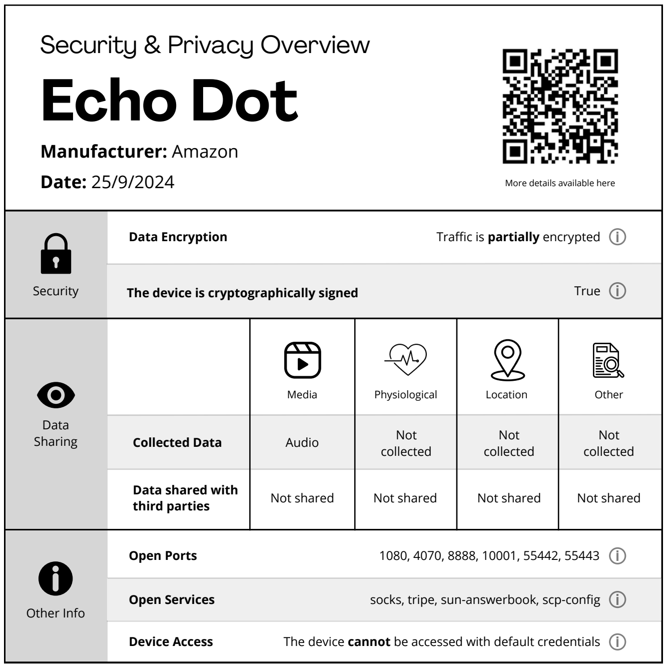
The list of open ports and services for the Amazon Echo Dot is shown in Table A. In contrast, the Nmap scan for the Ring Doorbell revealed no listening services.

We manually analyzed every device, revealing that many uncommon ports are used to host different services. Also, standard ports such as 22, 80, or 443 were not found open in any of the devices. We found 6668, 8443, 9000, and 9999 among the most common open ports.

*2) Default Credentials:* The default credentials factor relies on the open ports discovered earlier. However, since none of the identified listening services were suitable for default credential scanning, we assumed that no device accepted default credentials. The tool's results also indicated that none of the devices were accessible using default credentials.

## C. Final Labels

Labels like the ones reported in Figure 12 and 13 can be generated at the end of the automated analysis. Every information displayed on the labels is generated by *SPIoT*. The design is inspired by the one proposed by Carnegie Mellon University researchers on their website iotsecurityprivacy.org. It consists of several factors we could extract automatically and a QR code linked to the manufacturer's webpage, where they provide an in-depth explanation of each feature.



Security & Privacy Overview

**Echo Dot**

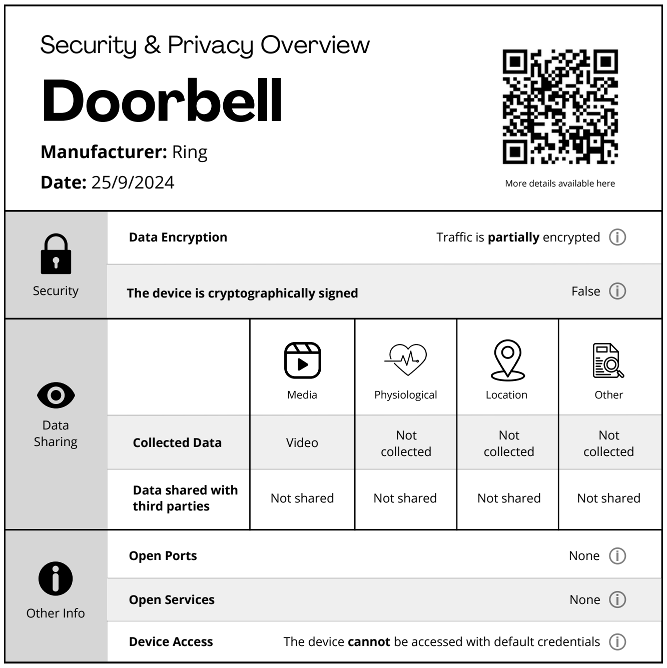
**Manufacturer:** Amazon

**Date:** 25/9/2024

More details available here

| | | | | | |
|---|---|---|---|---|---|
| Security | **Data Encryption** | Traffic is **partially** encrypted | | | |
| | **The device is cryptographically signed** | True | | | |
| Data Sharing | | Media | Physiological | Location | Other |
| | **Collected Data** | Audio | Not collected | Not collected | Not collected |
| | **Data shared with third parties** | Not shared | Not shared | Not shared | Not shared |
| Other Info | **Open Ports** | 1080, 4070, 8888, 10001, 55442, 55443 | | | |
| | **Open Services** | socks, tripe, sun-answerbook, scp-config | | | |
| | **Device Access** | The device **cannot** be accessed with default credentials | | | |

Fig. 12. Automatically generated label for the Amazon Echo Dot. The label provides an overview of the device's security and privacy features extracted using *SPIoT*. The label also features a QR code for users to access more detailed information from the manufacturer's webpage.



Security & Privacy Overview

**Doorbell**

**Manufacturer:** Ring

**Date:** 25/9/2024

More details available here

| | | | | | |
|---|---|---|---|---|---|
| Security | **Data Encryption** | Traffic is **partially** encrypted | | | |
| | **The device is cryptographically signed** | False | | | |
| Data Sharing | | Media | Physiological | Location | Other |
| | **Collected Data** | Video | Not collected | Not collected | Not collected |
| | **Data shared with third parties** | Not shared | Not shared | Not shared | Not shared |
| Other Info | **Open Ports** | None | | | |
| | **Open Services** | None | | | |
| | **Device Access** | The device **cannot** be accessed with default credentials | | | |

Fig. 13. Automatically generated label for the Ring Doorbell. The label provides an overview of the device's security and privacy features extracted using *SPIoT*.

## VII. Results and Discussion

The evaluation demonstrated the effectiveness of the proposed methodology in automating the passive and active analysis of IoT device traffic. By comparing the results of manual extraction with those obtained through *SPIoT*, it was shown that our methodology can assist manufacturers during the self-declaration phase.

Regarding accuracy, *SPIoT* performed well in identifying and classifying unencrypted data across various IoT devices. The tool effectively recognized traffic patterns and correctly categorized the media data transmitted to the cloud. For other types of data, the tool's accuracy improved with more advanced language models. Specifically, LLaMa3.1-8B achieved an average accuracy of 0.57965, while LLaMa3.1-70B scored a significantly higher accuracy of 0.9908 in extracting and classifying sensitive data. This highlights the importance of using more powerful models to improve extraction and categorization tasks, and it supports the idea that employing an LLM can be effective for this purpose.

From an efficiency standpoint, our methodology offers substantial time-saving benefits compared to fully manual feature extraction. The automation of network traffic analysis allows for the rapid identification of security and privacy factors, making it a helpful starting point for manufacturers to generate their self-declarations. As discussed in previous sections, manual analysis is time-consuming and prone to human error, particularly in complex traffic patterns. *SPIoT*'s ability to simplify this process significantly reduces the time required for feature extraction, improving efficiency and accuracy.

It is important to note that the proposed methodology does not solely rely on *SPIoT*, but rather on its integration into the manufacturer's analysis process. *SPIoT* acts as an assistant, helping with feature extraction and label generation. However, the tool is not perfect, as it is prone to errors and may miss important information that should be included in the final label.

Overall, the evaluation results indicate that our methodology is valuable for automating IoT device traffic analysis and has strong potential for improving efficiency and accuracy in data extraction tasks. Continuous improvements and updates to the tool will be necessary to ensure it remains up-to-date in a rapidly changing environment.

## VIII. Limitations and Future Work

The proposed methodology demonstrates significant potential for assisting manufacturers in generating security and privacy labels. However, further work is needed to maximize its effectiveness.

One of the primary limitations of *SPIoT* is its reliance on the quality and complexity of the underlying language models. As observed in the evaluation, data extraction and classification accuracy improves with more powerful models like LLaMa3.1-70B. Training a custom model specifically for data extraction and classification processes could improve even more the tool's accuracy. The tested LLMs were trained for general purposes, limiting their ability to handle the specific patterns in IoT traffic data. A dedicated model, trained exclusively on IoT-related data, could better recognize protocol structures, device behaviors, and sensitive data types, leading to more precise and reliable results.

Another notable limitation is that the methodology relies solely on network analysis. While this approach effectively identifies data transmitted to the cloud and successfully extracts the relevant factors, it provides only a partial view of the IoT device's behavior. Exploring different types of analysis could be beneficial for automatically extracting additional factors and generating more comprehensive labels. For instance, firmware analysis would be ideal for analyzing the device on a deeper level and could help extract factors that cannot be identified through network analysis. This analysis could reveal important information related to the device's sensors, physical actuators, passwords, and supported standards. Additionally, companion app analysis offers another viable approach. While not every IoT device has a companion app, more than 70% of devices use one. This is a significant enough percentage to justify the value of analyzing these apps, which could help extract factors such as data-sharing frequency, data storage locations, special handling practices for children's data, and the overall frequency of data collection.

Finally, integrating *SPIoT* into manufacturers' workflows presents its own set of challenges. Although the tool is designed to assist in feature extraction and label generation, it is not a fully autonomous solution. Human oversight is still required to verify the results and ensure that important details are not missed. Future work could focus on improving the integration of *SPIoT* into the label-generation process. For example, developing user-friendly interfaces or dashboards could make it easier for manufacturers to interact with the tool and monitor the results more effectively.

## IX. Conclusion

In this paper, we presented a novel methodology for automatically extracting features from IoT devices to generate privacy and security labels. By comparing the results of this automated process with manual extraction, we demonstrated that our methodology can significantly improve feature extraction efficiency, while also achieving high accuracy. However, further exploration of additional approaches, such as firmware and companion app analysis, is necessary to enhance the methodology. Extracting a higher number of features will enable the generation of more comprehensive privacy and security labels, leading to a deeper understanding of IoT device behavior and reinforcing consumer privacy and security awareness.

## References

[1] Y. Acar, M. Backes, S. Fahl, S. Garfinkel, D. Kim, M. L. Mazurek, and C. Stransky, "Comparing the usability of cryptographic APIs," in *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 154–171. [Online]. Available: http://ieeexplore.ieee.org/document/7958576/

[2] 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)*. San Francisco, CA, USA: IEEE, May 2019, pp. 1362–1380. [Online]. Available: https://ieeexplore.ieee.org/document/8835392/

[3] R. Balebako, A. Marsh, J. Lin, J. Hong, and L. Faith Cranor, "The Privacy and Security Behaviors of Smartphone App Developers," in *Proceedings 2014 Workshop on Usable Security*. San Diego, CA: Internet Society, 2014. [Online]. Available: https://www.ndss-symposium.org/ndss2014/workshop-usable-security-usec-2014-programme/privacy-and-security-behaviors-smartphone-app-developers

[4] M. Belshe, R. Peon, and M. Thomson, "Hypertext Transfer Protocol Version 2 (HTTP/2)," RFC Editor, Tech. Rep. RFC7540, May 2015. [Online]. Available: https://www.rfc-editor.org/info/rfc7540

[5] G. Chu, N. Apthorpe, and N. Feamster, "Security and Privacy Analyses of Internet of Things Children's Toys," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 978–985, Feb. 2019, arXiv:1805.02751 [cs]. [Online]. Available: http://arxiv.org/abs/1805.02751

[6] Creative Commons. CC0 1.0 Universal (CC0 1.0) Public Domain Dedication. [Online]. Available: https://creativecommons.org/publicdomain/zero/1.0/

[7] P. Emami-Naeini, Y. Agarwal, and L. F. Cranor. (2021) Specification for cmu iot security and privacy label (cispl 1.0). [Online]. Available: https://www.iotsecurityprivacy.org/downloads/Privacy_and_Security_Specifications.pdf

[8] P. Emami-Naeini, Y. Agarwal, L. Faith Cranor, and H. Hibshi, "Ask the experts: What should be on an IoT privacy and security label?" in *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2020, pp. 447–464. [Online]. Available: https://ieeexplore.ieee.org/document/9152770/

[9] P. Emami-Naeini, J. Dheenadhayalan, Y. Agarwal, and L. F. Cranor, "Which privacy and security attributes most impact consumers' risk perception and willingness to purchase IoT devices?" in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 519–536, ISSN: 2375-1207. [Online]. Available: https://ieeexplore.ieee.org/document/9519463

[10] P. Emami-Naeini, H. Dixon, Y. Agarwal, and L. F. Cranor, "Exploring how privacy and security factor into IoT device purchase behavior," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, ser. CHI '19. Association for Computing Machinery, 2019, pp. 1–12. [Online]. Available: https://dl.acm.org/doi/10.1145/3290605.3300764

[11] European Commission. EU-US Joint Statement on CyberSafe Products Action Plan - Shaping Europe's digital future. [Online]. Available: https://digital-strategy.ec.europa.eu/en/library/eu-us-joint-statement-cybersafe-products-action-plan

[12] ——, *EU Legislation in Progress - Cyber-Resilience Act*, 3rd ed. European Commission, 2023. [Online]. Available: https://www.europarl.europa.eu/RegData/etudes/BRIE/2022/739259/EPRS_BRI(2022)739259_EN.pdf

[13] European Parliament. Procedure File: 2017/2068(INI) | Legislative Observatory | European Parliament. [Online]. Available: https://oeil.secure.europarl.europa.eu/oeil/popups/ficheprocedure.do?lang=en&reference=2017/2068(INI)

[14] I. Hadar, T. Hasson, O. Ayalon, E. Toch, M. Birnhack, S. Sherman, and A. Balissa, "Privacy by designers: software developers' privacy mindset," *Empirical Software Engineering*, vol. 23, no. 1, pp. 259–289, Feb. 2018. [Online]. Available: http://link.springer.com/10.1007/s10664-017-9517-1

[15] D. Y. Huang, N. Apthorpe, G. Acar, F. Li, and N. Feamster, "IoT Inspector: Crowdsourcing Labeled Network Traffic from Smart Home Devices at Scale," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4, no. 2, pp. 1–21, Jun. 2020, arXiv:1909.09848 [cs]. [Online]. Available: http://arxiv.org/abs/1909.09848

[16] M. A. Jan, F. Khan, M. Alam, and M. Usman, "A payload-based mutual authentication scheme for Internet of Things," *Future Generation Computer Systems*, vol. 92, pp. 1028–1039, Mar. 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0167739X17303898

[17] Kaspersky, "Black Hat USA 2015: The full story of how that Jeep was hacked," Aug. 2015. [Online]. Available: https://www.kaspersky.com/blog/blackhat-jeep-cherokee-hack-explained/9493/

[18] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: Mirai and Other Botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7971869/

[19] D. Kumar, K. Shen, B. Case, D. Garg, G. Alperovich, D. Kuznetsov, R. Gupta, and Z. Durumeric, "All things considered: An analysis of IoT devices on home networks," in *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, Aug. 2019, pp. 1169–1185. [Online]. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/kumar-deepak

[20] M. Labs, "150,000 Verkada security cameras hacked—to make a point," Mar. 2021. [Online]. Available: https://www.malwarebytes.com/blog/news/2021/03/150000-verkada-security-cameras-hacked-to-make-a-point

[21] V. Lara, "What the Internet of Things means for consumer privacy." [Online]. Available: https://impact.econ-asia.com/perspectives/technology-innovation/what-internet-things-means-consumer-privacy-0/white-paper/what-internet-things-means-consumer-privacy

[22] T. Li, Y. Agarwal, and J. I. Hong, "Coconut: An IDE Plugin for Developing Privacy-Friendly Apps," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 2, no. 4, pp. 1–35, Dec. 2018. [Online]. Available: https://dl.acm.org/doi/10.1145/3287056

[23] T. Li, L. F. Cranor, Y. Agarwal, and J. I. Hong, "Matcha: An IDE plugin for creating accurate privacy nutrition labels." [Online]. Available: http://arxiv.org/abs/2402.03582

[24] T. Li, K. Reiman, Y. Agarwal, L. F. Cranor, and J. I. Hong, "Understanding challenges for developers to create accurate privacy nutrition labels," in *CHI Conference on Human Factors in Computing Systems*. ACM, 2022, pp. 1–24. [Online]. Available: https://dl.acm.org/doi/10.1145/3491102.3502012

[25] F. Loi, A. Sivanathan, H. H. Gharakheili, A. Radford, and V. Sivaraman, "Systematically Evaluating Security and Privacy for Consumer IoT Devices," in *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy*. Dallas Texas USA: ACM, Nov. 2017, pp. 1–6. [Online]. Available: https://dl.acm.org/doi/10.1145/3139937.3139938

[26] M. Mainuddin, Z. Duan, and Y. Dong, "Network Traffic Characteristics of IoT Devices in Smart Homes," in *2021 International Conference on Computer Communications and Networks (ICCCN)*. Athens, Greece: IEEE, Jul. 2021, pp. 1–11. [Online]. Available: https://ieeexplore.ieee.org/document/9522168/

[27] S. Marchal, "IoT devices captures," 2017, artwork Size: 25MB Pages: 25MB. [Online]. Available: https://research.aalto.fi/en/datasets/iot-devices-captures(285a9b06-de31-4d8b-88e9-5bdba46cc161).html

[28] M. H. Mazhar and Z. Shafiq, "Characterizing Smart Home IoT Traffic in the Wild," Mar. 2020, arXiv:2001.08288 [cs]. [Online]. Available: http://arxiv.org/abs/2001.08288

[29] W. R. F. Merkel, "DRAFT Baseline Security Criteria for Consumer IoT Devices," *Cybersecurity White Paper*, 2021. [Online]. Available: https://www.nist.gov/system/files/documents/2021/08/31/IoT%20White%20Paper%20-%20Final%202021-08-31.pdf

[30] Meta, "LLaMa3," https://github.com/meta-llama/llama3, 2024.

[31] ——, "LlamaAPI Documentation," https://docs.llama-api.com/quickstart, 2024.

[32] D. Miessler, "SecLists," https://github.com/danielmiessler/SecLists/tree/master/Passwords/Default-Credentials, 2020.

[33] ——, "SecLists," https://github.com/danielmiessler/SecLists/tree/master/Passwords/Default-Credentials, 2024.

[34] National Institute of Standards and Technology, "Cybersecurity White Paper: EO Response," National Institute of Standards and Technology, Tech. Rep., 2022. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.02042022-2.pdf

[35] Nigel Cory, "Why the United States and EU Should Seize the Moment to Cooperate on Cybersecurity Labeling for IoT Devices," ITIF - Information Technology & Innovation Foundation, Tech. Rep., 2024. [Online]. Available: https://itif.org/publications/2024/03/28/why-us-eu-should-cooperate-on-cybersecurity-labeling-for-iot-devices/

[36] R. K. Niven, M. Abel, M. Schlegel, and S. H. Waldrip, "Maximum Entropy Analysis of Flow Networks: Theoretical Foundation and Applications," *Entropy*, vol. 21, no. 8, p. 776, Aug. 2019. [Online]. Available: https://www.mdpi.com/1099-4300/21/8/776

[37] Ollama, "Ollama," https://github.com/ollama/ollama, 2023.

[38] S. Shasha, M. Mahmoud, M. Mannan, and A. Youssef, "Playing With Danger: A Taxonomy and Evaluation of Threats to Smart Toys," *IEEE*

*Internet of Things Journal*, vol. 6, no. 2, pp. 2986–3002, Apr. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8502818/

[39] I. Spectrum, "The Real Story of Stuxnet - IEEE Spectrum." [Online]. Available: https://spectrum.ieee.org/the-real-story-of-stuxnet

[40] T. Wu, F. Breitinger, and S. Niemann, "IoT network traffic analysis: Opportunities and challenges for forensic investigators?" *Forensic Science International: Digital Investigation*, vol. 38, p. 301123, Oct. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2666281721000214

## Appendix

TABLE II
Categories of Sensitive Data

| **Category** | **Sensitive Data** |
|---|---|
| **Media** | Audio, Video, Image |
| **Physiological** | Height, Weight, Age, Gender, Heart rate, Blood pressure, Respiratory rate, Body temperature, Steps taken, Calories burned, Exercise intensity, Blood glucose level, Oxygen saturation, Stress level, Mood indicator, Fatigue level |
| **Location** | Latitude, Longitude |
| **Other Data** | IP address, MAC address, Temperature, Humidity, Light level, Air quality, Pressure, Battery level, Device info, Firmware version, Username, Password |

TABLE III
Comprehensive list of every data found by manually inspecting the D-Link HD IP Camera DCH-935L

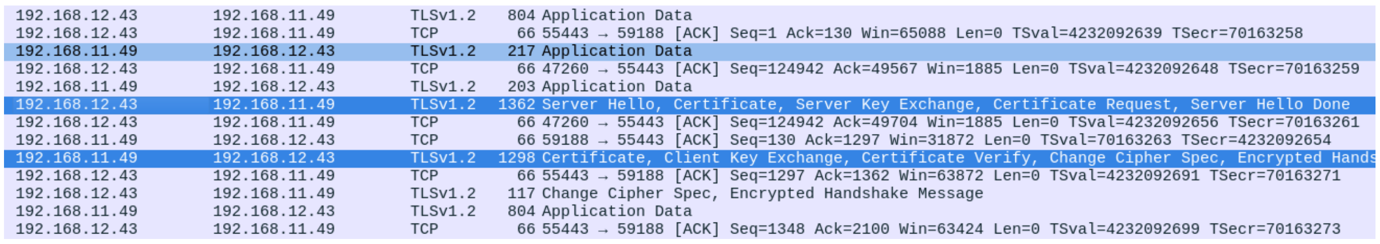
| **Field** | **Value** |
|---|---|
| LoginResult | success |
| Challenge | A5F7d380b830436c59eA |
| Cookie | 5ac3752C84 |
| PublicKey | 57d56bA23740B9e14713 |
| GetDeviceSettingsResult | OK |
| Type | ConnectedHomeClient |
| DeviceName | DCS-935L |
| VendorName | D-Link |
| ModelDescription | Wireless Internet Camera |
| ModelName | DCS-935L |
| DeviceMacId | B0:C5:54:25:5B:0E |
| FirmwareVersion | 1.06 |
| FirmwareRegion | Default |
| LatestFirmwareVersion | 1.06 |
| HardwareVersion | A1 |
| HNAPVersion | 0111 |
| PresentationURL | http://dcs-935l.local./ |
| CAPTCHA | false |
| string | en-HK |
| SubDeviceURLs | null |
| SetDeviceSettingsResult | OK |
| GetDeviceSettings2Result | OK |
| SerialNumber | 0 |
| TimeZone | 1 |
| AutoAdjustDST | true |
| Locale | en-GB |
| SSL | true |
| SetDeviceSettings2Result | OK |
| GetmydlinkRegInfoResult | OK |
| mydlinknumber | 32779615 |
| Macaddress | B0:C5:54:25:5B:0E |
| Footprint | 6BE6E73A4A78923A5376FCA56C8B37 |
| Registered | false |
| SetmydlinkRegResult | OK |

TABLE IV
List of open ports and services for the Amazon Echo Dot

| **Port** | **State** | **Service** |
|---|---|---|
| 1080/tcp | open | socks |
| 4070/tcp | open | tripe |
| 8888/tcp | open | sun-answerbook |
| 10001/tcp | open | scp-config |
| 55442/tcp | open | unknown |
| 55443/tcp | open | unknown |

TABLE V
mTLS Support in UT Lab IoT Devices

| **Device** | **Cryptographically Signed** |
|---|---|
| **Amazon Echo Dot** | **Yes** |
| Ring Doorbell | No |
| Samsung Fridge | No |
| Google Nest Cam | No |
| Bosch Dishwasher | No |
| Smartplug - KP105 | No |
| Smartplug - HS110 | No |
| SALCAR Radiator Thermostat TRV801W | No |
| Philips Air Purifier 600-Serie | No |
| **Philips TV** | **Yes** |
| Chromecast v3 | No |
| PNI Safe House PG600LR | No |
| Calex motion sensor | No |
| Philips HUE | No |
| Nest smoke alarm | No |
| Google Home Mini | No |
| Amazon Echo | No |
| Princess Smart Aerofryer | No |
| Calex Slimme LED Vloerlamp | No |
| Ilife t10s | No |
| WellToBe pet feeder | No |
| Watersensor | No |
| PS5 | No |



```
192.168.12.43        192.168.11.49        TLSv1.2   804 Application Data
192.168.12.43        192.168.11.49        TCP        66 55443 → 59188 [ACK] Seq=1 Ack=130 Win=65088 Len=0 TSval=4232092639 TSecr=70163258
192.168.11.49        192.168.12.43        TLSv1.2   217 Application Data
192.168.12.43        192.168.11.49        TCP        66 47260 → 55443 [ACK] Seq=124942 Ack=49567 Win=1885 Len=0 TSval=4232092648 TSecr=70163259
192.168.11.49        192.168.12.43        TLSv1.2   203 Application Data
192.168.12.43        192.168.11.49        TLSv1.2  1362 Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
192.168.12.43        192.168.11.49        TCP        66 47260 → 55443 [ACK] Seq=124942 Ack=49704 Win=1885 Len=0 TSval=4232092656 TSecr=70163261
192.168.11.49        192.168.12.43        TCP        66 59188 → 55443 [ACK] Seq=130 Ack=1297 Win=31872 Len=0 TSval=70163263 TSecr=4232092654
192.168.11.49        192.168.12.43        TLSv1.2  1298 Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Hands
192.168.12.43        192.168.11.49        TCP        66 55443 → 59188 [ACK] Seq=1297 Ack=1362 Win=63872 Len=0 TSval=4232092691 TSecr=70163271
192.168.12.43        192.168.11.49        TLSv1.2   117 Change Cipher Spec, Encrypted Handshake Message
192.168.11.49        192.168.12.43        TLSv1.2   804 Application Data
192.168.12.43        192.168.11.49        TCP        66 55443 → 59188 [ACK] Seq=1348 Ack=2100 Win=63424 Len=0 TSval=4232092699 TSecr=70163273
```

Fig. 14. mTLS between Amazon Echo Dot (192.168.12.43) and Amazon Echo (192.168.11.49).