Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 6731783, 16 pages
https://doi.org/10.1155/2023/6731783

Research Article

ZPA: A Smart Home Privacy Analysis System Based on ZigBee Encrypted Traffic

Rong Li,[1] **Wei Zhang**,[1] **Lifa Wu**,[1] **Yunfei Tang**,[2] **and Xinguang Xie**[1]

[1]*School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China*
[2]*Bell College of Talents, Nanjing University of Posts and Telecommunications, Nanjing 210023, China*

Correspondence should be addressed to Lifa Wu; wulifa@njupt.edu.cn

Received 6 June 2022; Revised 15 October 2022; Accepted 25 November 2022; Published 31 January 2023

Academic Editor: Kuruva Lakshmanna

Currently, the ZigBee protocol is widely used in smart homes and provides convenience to people. However, smart home devices often carry a large amount of real physical world information, which may result in information leakage problems. In this paper, to reveal the privacy security issues existing in ZigBee-based smart home networks, we design a smart home privacy analysis system based on ZigBee-encrypted traffic, called ZPA. ZPA can extract ZigBee data features based on the device's operating mode and time window and use state-of-the-art machine-learning models to identify the type and status of smart home devices that could leak users' private information. Through the analysis of 20 different devices from 5 manufacturers, the results show that even if the ZigBee traffic is protected by encryption, the accuracy of the proposed method in device type identification and state inference can reach approximately 93% and 98%, respectively. The types and statuses of devices in smart homes will reveal the user's activity information to a certain extent. The privacy security of ZigBee-based smart devices still needs to be further strengthened.

1. Introduction

In recent years, consumer IoT devices have entered a period of rapid development as consumers' demands for smart living continue to increase. According to the 2021 annual statistics [1] released by Xiaomi, the number of IoT devices will exceed 75 billion, the IoT market space will exceed 81 billion US dollars, and the IoT device data will reach 79.4 ZB by 2025. Different from traditional home appliances, most consumer smart home devices are equipped with wireless communication interfaces to realize interconnection between smart devices in the smart home and support users in remotely controlling the devices through the cloud or mobile apps. For example, users can remotely turn on smart light bulbs through a mobile app.

Although the extensive use of IoT devices improves people's lives, it brings information security risks. Many smart home devices continuously collect information from the physical world that carries a large amount of user activity information and even some private user information. The sensitive information is leaked out without the user's awareness. The communication interfaces and control functions allow attackers to obtain communication traffic between smart home devices and even type and status information of the device. Due to the slow iteration cycle of smart devices, smart home devices cannot patch security vulnerabilities in time. After identifying the specific device type, attackers can find smart home devices with known vulnerabilities to develop targeted attack schemes to control the target device, such as the unauthorized opening of intelligent door locks and powering smart doorbells and smart cameras on and off. Since the states of smart devices are generally limited, attackers can further infer the status of a specific device through traffic analysis and infer user behaviors through changes in smart device states. For example, an attacker can analyze the user's life rules through the status change of a door sensor or infer whether there is someone at home through the status changes of a human body sensor.

These leaked smart home device types and status information pose a severe threat to users' personal and property security.

In response to the privacy issue of smart homes, many researchers have begun to study this issue. Most of the existing research on identifying the types and statuses of smart home devices is based on obtaining device traffic from home wireless routers. This scenario has certain limitations for attackers, who cannot easily monitor the communication traffic of users' home routers. In addition, the router has further processed the device traffic it captured; thus, some important data characteristics may be lost. However, Wi-Fi is not the only communication protocol used in smart homes, and small smart devices often use ZigBee protocols to communicate with routers to save power.

ZigBee is widely used in smart home devices as a low-cost and low-power wireless communication protocol. Due to its openness and low complexity, attackers can passively capture device traffic data directly over the air without obtaining device traffic information from the wireless router. Attackers can easily obtain ZigBee device traffic and analyze user privacy information. Therefore, in this paper, we design a smart home privacy detection system called ZPA that is based on ZigBee-encrypted traffic. Through experiments on multiple similar devices from multiple manufacturers, we are able to identify the type and status information of smart devices with high accuracy, which reveals the privacy security problems existing in ZigBee-based smart homes.

The contributions of this paper mainly include the following:

(1)In our experiment, we found that all manufacturers' smart home ZigBee devices have improper key usage. These devices all use the default global link key. An attacker can decrypt the initial traffic of ZigBee devices and obtain the network key in the ZigBee network. Thus, attackers have been able to decrypt all data traffic in ZigBee networks, resulting in leakage of user privacy

(2)We propose a smart home device type identification method based on ZigBee-encrypted traffic. By analyzing the unique operation mode of the smart device, we divide traffic into the form of time windows, construct device-dependent feature vectors, and use the machine learning classification model to build the system. Adding the hard-code features and heartbeat packet statistics can significantly improve the recognition rate of device types among different manufacturers. Experiments show that even if the ZigBee network traffic of the smart home is encrypted, the recognition system can recognize the device type from multiple manufacturers with an accuracy of approximately 93%

(3)We propose a smart home device state identification method based on the device type. We divide the devices into three classes according to the differences between the states of smart devices. On the basis of identifying specific device types, we establish a state recognition classification model for each device. Experiments show that the recognition system can identify specific device states with a high accuracy of approximately 98%

The rest of this paper is organized as follows. We provide the background of the ZigBee protocol in Section 2 and review the related work in Section 3. We describe attack scenarios in Section 4 and design a ZigBee device privacy detection system in Section 5. The privacy and security issues of ZigBee devices during initialization are presented in Section 6. Section 7 details the implementation of smart home device type and status recognition, and the experimental results are presented in Section 8. We discuss the defensive measures and some related issues in Section 9 and finally conclude the paper in Section 10.

2. Background

In this section, we briefly introduce the basic information of the ZigBee protocol, including the protocol structure, device types, and security measures in ZigBee networks.

2.1. ZigBee Structure. ZigBee is a low-cost, low-power wireless communication protocol mainly used in embedded devices for short-distance communication. The ZigBee protocol is based on the IEEE 802.15.4 standard and is supported by the ZigBee Alliance [2].

The ZigBee protocol architecture is shown in Figure 1. The ZigBee protocol architecture consists of the following: the physical layer (PHY), the media access control layer (MAC), the network layer (NWK), and the application layer (APL). The physical layer provides the most basic services, such as data interfaces. The MAC layer is responsible for the establishment, maintenance, termination, and confirmed data transmission and reception of wireless data links between different devices. These two layers are defined by the IEEE 802.15.4 standard. ZigBee builds NWK and APL layers on top of the PHY and MAC layers and uses the AES-CCM* algorithm to encrypt the contents. The NWK layer is responsible for securely transmitting outgoing and incoming frames. The application layer handles device application commands, including the Application Support Sublayer (APS), ZigBee Device Object (ZDO), and application framework.

2.2. ZigBee Device Types. A ZigBee network consists of the following three devices: coordinator, end device, and router. ZigBee coordinator is mainly responsible for establishing, implementing, and managing the entire ZigBee network. The ZigBee end device is usually a sensor node device used to monitor and collect environmental data. The ZigBee router is an intermediate node device responsible for transmitting data packets. In the ZigBee network of a typical smart home, only the coordinator and end devices are generally included, as shown in Figure 2.

Each device in a ZigBee network has the following two addresses: the MAC address and the short address. The MAC address is unique and is used as the sender and receiver's address for data transmission during the initial process. The short address is two bytes in size and is assigned to the device by the coordinator after the device connects to the network. After the device is connected to the network, it uses the short address to communicate with other devices.

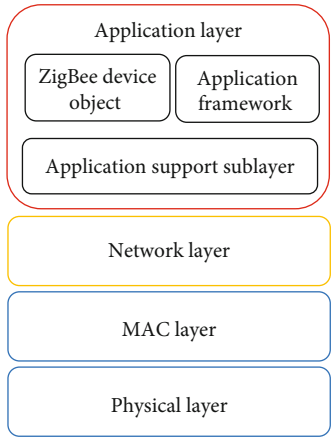

Figure 1: ZigBee protocol architecture.

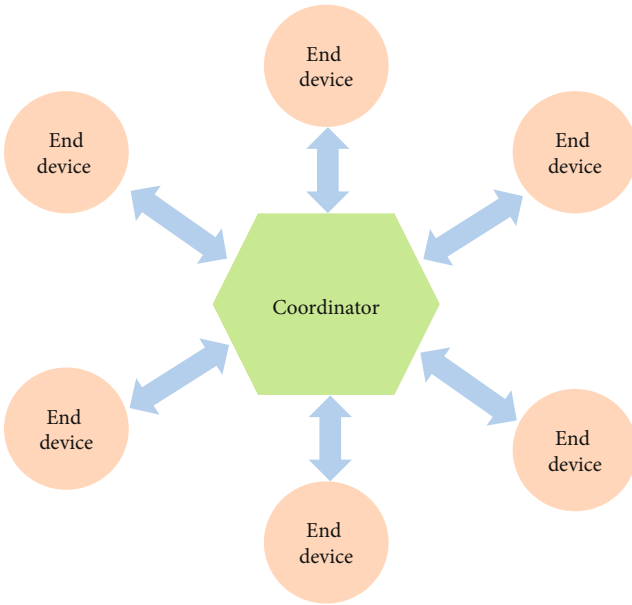

Figure 2: ZigBee network topology for a typical smart home.

2.3. ZigBee Security Measures. To ensure the security and usability of the device, ZigBee provides the following two types of security models: distributed and centralized. The distributed ZigBee network has two device types, i.e., router and end devices, where each router can issue a network key. The centralized security model is more complicated and secure than the distributed security model. The centralized ZigBee network is formed by the Trust Center (TC), which is responsible for distributing keys to the devices in the network.

In the ZigBee network, two keys are used to protect the security of messages: the link key and the network key. These two keys are used to encrypt data at different stages in the ZigBee network. There are two types of the link keys, as follows: a global key and unique key. The global key exists in the TC and each device in the network and is mainly used to encrypt the negotiated data during the network access process of the device. The unique link key is mainly used for a one-to-one relation between the TC and an end device or a pair of end devices. The network key is also a 128-bit key shared by end devices in a network, which is sent to each end device by the TC and is mainly used for information encryption in the network layer.

3. Related Work

With the continuous development of IoT technology, researchers have paid more attention to the privacy and security issues of IoT devices. To explore the security of smart home devices, it is first necessary to establish a device fingerprint identification model. There have been some works on IoT device fingerprint identification proposed in recent years, but they use only a small number of devices [3–5] or the level of fingerprint identification is coarse-grained [6–10]. To prove the universality of device fingerprinting, Dalai and Jena [11] proposed a method for fingerprint identification of wireless devices that uses the homogeneity of devices of the same brand and the heterogeneity of devices of different brands to create unique and replicable device signatures. Three hundred device types and over 1.5 GB of network traffic were used for model analysis and performance evaluation. Shahid et al. [12] proposed a method to identify the types of IoT devices connected to the network. They analyzed the packet traffic sent and received, visualized the data by using T-SNE technology, and then used six different machine learning classifiers to identify IoT devices. To further describe the device behavior and process encrypted traffic, Bezawada et al. [13] proposed a device fingerprint recognition method that can be employed to undertake device type identification. Because the above research assumes the presence of local attackers or a simple network environment, without considering whether traffic analysis is practical in more complex networks, Dong et al. [14] assumed that gateways might enable standard obstructive traffic analysis configurations, such as Virtual Private Network (VPN) and Network Address Port Translation (NAPT). By analyzing IoT device traffic in real-world and public datasets, a traffic analysis framework based on sequence learning technology such as Long Short-Term Memory (LSTM) is proposed for device fingerprint recognition.

Based on device fingerprint identification, network traffic analysis has been shown to be reasonably effective when applied to anomaly detection [15–20], IoT device privacy leakage [14, 21, 22] and IoT device privacy protection [23, 24]. For anomaly detection, Miettinen et al. [15] designed an IoT Sentinel framework for device fingerprint identification and IoT network protection. When devices are first registered on the network, the framework uses machine learning methods to fingerprint them and limits the communication of vulnerable devices to minimize risk. Mirsky et al. [19] presented a plug and play NIDS that uses neural networks to detect attacks on a local network without supervision, and first proposed the use of autoencoders with or without ensembles for online anomaly detection. Facing the problem at different levels of granularity, Bovenzi et al. [20] proposed a hierarchical hybrid intrusion detection

Table 1: Description of the related fingerprinting methods.

Approach	Identification type	Features	Technique	Purpose
Corbett et al. [3]	NIC vendor	Traffic rate	Spectral analysis	Protects networks
Bratus et al. [4]	Chipsets and drivers	Response to crafted frames	Decision list learning	Identifies fake APs
Gao et at. [5]	APs	Interarrival time	Wavelet analysis	Detects unsafe APs
Loh et al. [8]	802.11 devices	Probe request frames	Timing analysis	Protects networks
Neumann et al. [10]	802.11 devices	Transmission time and interarrival time	Histogram and cosine similarity	Device fingerprinting
Dalai and Jena[11]	Wi-Fi devices	Correlation-based feature selection	Similarity measure	Device fingerprinting
Shahid et al. [12]	Wi-Fi devices	Packet size and time	T-SNE technology, machine learning	Device behavior description
Bezawada et al. [13]	Wi-Fi devices	Packet header and payload features	Machine learning	Device type identification
Dong et al. [14]	Wi-Fi devices	Packet header and payload features	LSTM	Device type identification
Mirsky et al. [19]	IP camera	Temporal statistics features	Unsupervised ANN	Anomaly detection
Bovenzi et al. [20]	Wi-Fi devices	TCP/IP stack layer features	Deep autoencoders, machine learning, and double-censoring mechanism	Anomaly detection
Acar et al. [22]	Wi-Fi, BLE, and ZigBee devices	Timing features, sensor state and controller state features, and controller location features	Machine learning	Device privacy leakage
Singh et al. [23]	Wi-Fi-based wireless sensors	MAC address, cause-effect relationship	Granger causality, dead reckoning	Device privacy protection

approach. For IoT device privacy leakage, Acar et al. [22] proposed a multistage privacy attack in an intellectual environment, which acquired the network traffic of smart home devices through passive sniffing, including Wi-Fi, Bluetooth Low Energy (BLE), and ZigBee devices. Machine learning methods are used to detect and identify IoT device types, states, and ongoing user activities. At the same time, the author also proposed a strategy based on generating spoofing traffic to hide the device state and provide better protection for smart home devices. For IoT device privacy protection, Singh et al. [23] proposed a framework called SNOOPDOG that not only detects common Wi-Fi-based wireless sensors that are actively monitor a user, but also classifies and localizes each device. A detailed description of some existing device fingerprint identification methods is shown in Table 1.

Although a variety of smart home devices have been used in device fingerprinting studies, but most use Wi-Fi routers to collect device traffic. Our work mainly aims at ZigBee devices used in the Internet of Things, where device traffic does not need to be captured through routers. The raw traffic data of ZigBee devices can be obtained by direct airborne packet capture, which is also easier to implement. This is also an aspect that is often overlooked in ZigBee privacy security.

4. Attack Scenarios

There are three common wireless protocol traffic types in a typical smart home environment, namely, Wi-Fi, BLE, and ZigBee traffic. For the Wi-Fi protocol, because the communication data are protected by Wi-Fi protected access (WPA/WPA2/WPA3) and Transport Layer Security (TLS), it is very difficult for an attacker to directly analyze smart home devices through encrypted data packets without knowledge of the wireless passwords. There are some ways to crack Wi-Fi passwords, but they consume too many resources and are not discussed in this article. In addition, Wi-Fi traffic data contain a large amount of mobile phone and host traffic. Identifying smart home traffic from large data traffic is also a challenge. For BLE traffic, it is challenging to capture complete traffic during data transmission because devices use a frequency-hopping algorithm to hop between data channels. Incomplete data can mislead classification models, causing them to provide false results. For small smart home devices, the ZigBee protocol is a better choice. The ZigBee protocol is simple and has low power consumption, which can prolong the use time of smart devices, but it is also a good target for attackers.

For the conventional smart home scenario, we design the attack scenario in Figure 3, which shows how the attacker can identify the type and status of devices in the smart home without directly touching them.

In our attack scenario, the smart home devices continuously collect environmental information about the home's environment and transmit the data to the smart gateway using the ZigBee protocol. The smart gateway acts as a coordinator. The smart gateway is responsible for managing the ZigBee network and transmits environmental information to the cloud server of the manufacturer through a Wi-Fi router. Users can not only check the environmental information monitored by smart home devices through mobile

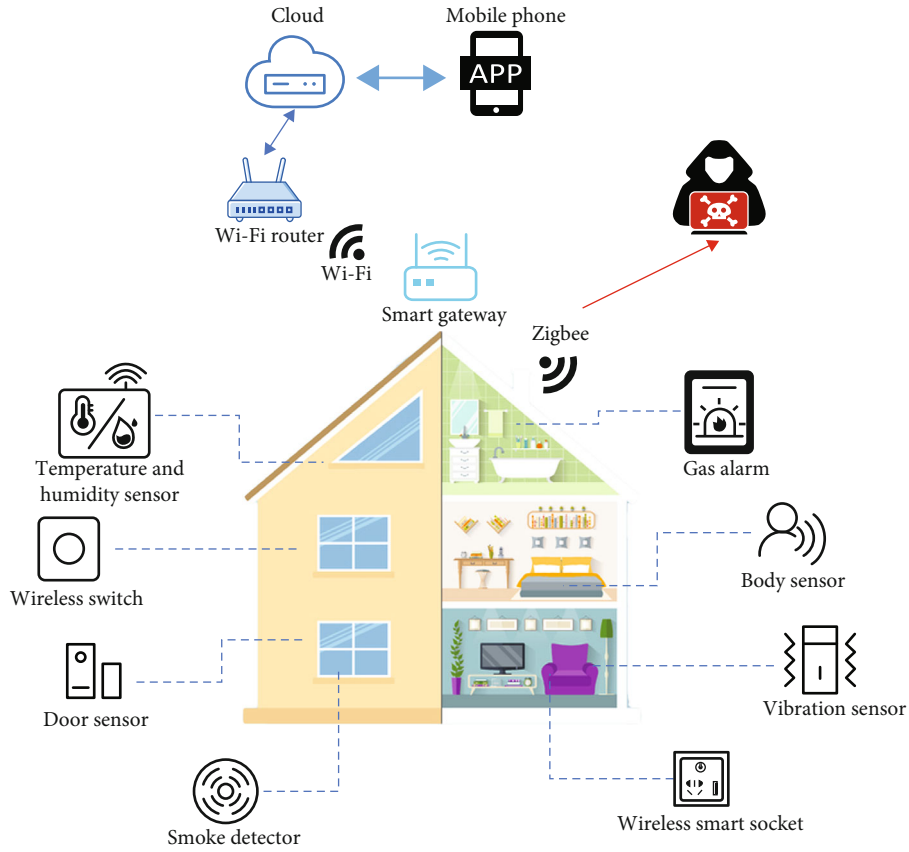

Figure 3: Privacy detection attack scenario.

phone apps but can also remotely control smart home devices and customize some intelligent rules. For example, a user can remotely turn on a wireless smart socket and power a device connected to the socket.

The environment information of smart homes will exist in ZigBee and Wi-Fi traffic. In our study, attackers only need to focus on ZigBee traffic to steal user home environment information. Since the working channel of ZigBee devices will not change, attackers only need a simple device to capture the traffic data of ZigBee-based smart home devices in the air. After obtaining the device communication traffic, the attacker analyzes the communication traffic of smart devices through the smart device classification system, infers the smart sensors and their status in the user's home, and even further infers the user's behavior.

5. Privacy Detection System Design

Based on the attack scenario discussed in the previous section, we design a smart home privacy detection system called ZPA that is based on ZigBee-encrypted traffic. The system structure is shown in Figure 4.

In the designed system, we first passively collect the data traffic of smart devices through air packet capture. Then, we preprocess the ZigBee network traffic and extract the traffic packet of each smart device according to the short address. Depending on the device traffic captured, attackers have two ways to identify the types and statuses of smart home devices. In the first scenario, the attacker is able to capture the initial traffic of the device. The results of our experiment show that all ZigBee smart devices use the default global link key, which is used to encrypt the network key transmission process during the network access process of smart devices. The ZigBee network key can be obtained by the attacker by decrypting the initial traffic of the device. Because the environmental information of ZigBee devices is encrypted using network keys, the attacker can directly decrypt packets to obtain the environmental information of smart homes.

For another scenario in which the attacker cannot obtain the device's traffic in the initialization process, the device type and state characteristics are extracted for each smart device. The device type and state characteristics are entered into the classification model for training to construct a smart home device privacy detection system. Even if the initialized traffic cannot be obtained and the data are encrypted, the type and status of the smart device can be identified.

5.1. Data Collection. To demonstrate the performance of our multimanufacturer identification model on similar devices, we first select some smart home device categories related to user behavior and then choose some mainstream smart home brands and ZigBee devices with similar functions according to market sales, including the following:

(1) Xiaomi: smoke detector, body sensor, gas alarm, and smart gateway

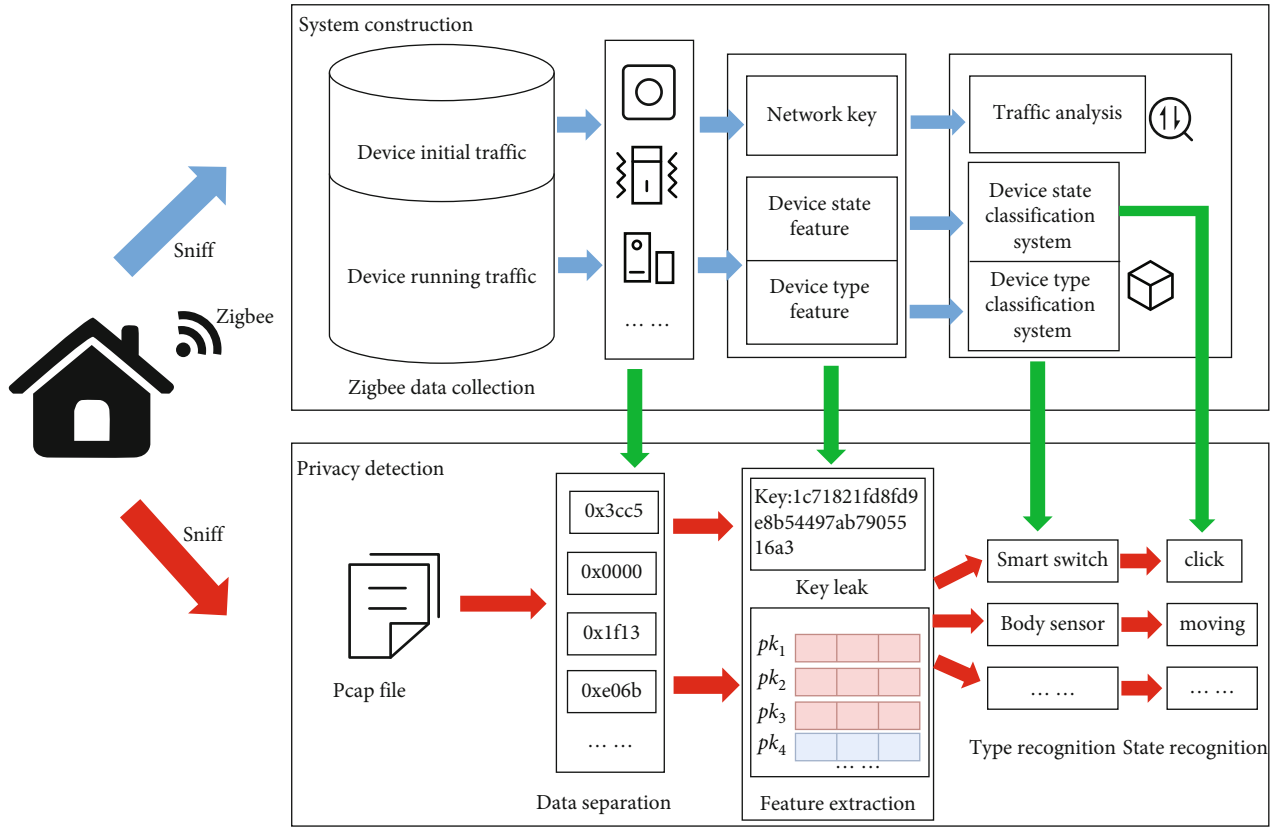

Figure 4: Smart home device privacy detection system ZPA.

(2) Jingdong: body sensor, wireless switch, door sensor, smart wireless socket, and smart gateway

(3) Aqara: door sensor, vibration sensor, body sensor, and smart gateway

(4) Haiman: body sensor, door sensor, and smart gateway

(5) Huawei: temperature and humidity sensor, door sensor, body sensor, and smart gateway

We built a ZigBee-based smart device environment in the lab. Before capturing the device traffic, we need to test each ZigBee channel to determine the specific working channel of the smart device. We use a CC2531 USB dongle as the hardware to capture the device traffic and collect the network traffic of 20 different smart devices from the above five manufacturers. To collect a more comprehensive data flow from each smart device and show the unique operating characteristics of each smart device, we divide the data collection into two parts. In the first part, smart devices are directly deployed in the laboratory for 1-2 days to simulate a real user environment to collect traffic data that is as close to the real situation as possible; the second part involves manually triggering the smart device and repeating this step more than 20 times to capture more device trigger data.

The numbers and sizes of the captured ZigBee packets from various manufacturers are shown in Table 2.

Table 2: The traffic packet size and number of each manufacturer.

Manufacturer	Size (MB)	Number
Xiaomi	4.5	40463
Jingdong	2.22	19559
Aqara	0.57	5269
Haiman	1.9	16291
Huawei	5.3	43979

5.2. Data Separation. In the ZigBee smart home network, devices from different manufacturers may communicate through different ZigBee channels. A ZigBee channel has multiple Personal Area Network ID (PAN ID) networks generated by a smart gateway, and there are multiple smart devices in one network. Each device has a unique short address to transmit data in the network. We separate the data according to the short address of the device and divide the overall ZigBee network traffic packet into the independent traffic of each device. Since each manufacturer needs a smart gateway to act as the coordinator in the network and is responsible for the management function of the whole ZigBee network when establishing the ZigBee network, its short address is generally 0x0000. It is easy to distinguish the traffic packets sent by the smart gateway in the ZigBee network. Therefore, we remove the smart gateway device traffic from the complete ZigBee network traffic and then divide the remaining traffic according to each subdevice.

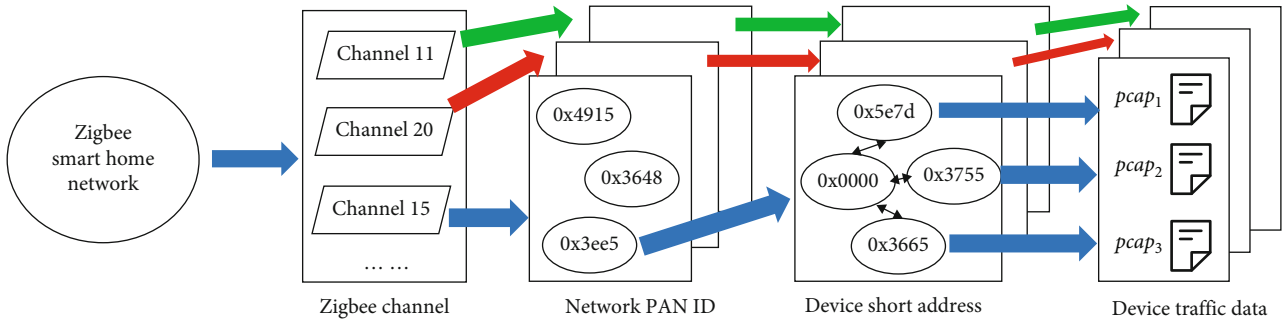

Figure 5: ZigBee network traffic separation.

Table 3: Traffic packet size and number of subdevice of each manufacturer.

Subdevice	Size (KB)	Number	Percentage
Jingdong-wireless smart socket	737	6312	13.03%
Jingdong-body sensor	144	1423	2.94%
Jingdong-wireless switch	13	128	0.26%
Jingdong-door sensor	34	381	0.79%
Xiaomi-smoke detector	785	9276	19.15%
Xiaomi-body sensor	107	1024	2.11%
Xiaomi-gas alarm	1910	16102	33.24%
Aqara-door sensor	69	666	1.38%
Aqara-vibration sensor	41	381	0.79%
Aqara-body sensor	286	2710	5.59%
Haiman-body sensor	383	4128	8.52%
Haiman-door sensor	124	1316	2.71%
Huawei-body sensor	234	2166	4.47%
Huawei-door sensor	98	1058	2.18%
Huawei-temperature and humidity sensor	128	1358	2.84%

Each piece of network traffic is regarded as a sequence of data packets constructed by the two entities in chronological order to facilitate the feature processing of subsequent device traffic. Compared with IP-routing traffic, ZigBee traffic data are relatively small. Only MAC layer data can be parsed when ZigBee data are encrypted with the network key. Although ZigBee network data use AES encryption to protect its payload, data encryption does not cover some basic characteristics of communication traffic, such as the length of the data packet, the timestamp of the data packet, and the short addresses of the communicating parties. At the same time, the encryption of the network layer cannot protect the MAC layer data. When the data packet sent by the communicating party uses the MAC layer command, it is easy to parse and obtain the content of the data packet. We use Python's Scapy library (the latest version is 2.4.5) [25] to manage the ZigBee network data traffic. The data separation process is shown in Figure 5.

Due to the different operating logics of devices from different manufacturers, there are certain differences in the number of ZigBee traffic packets we collect. For example, the gas alarm continuously sends broadcast packets, and since it is directly connected to a power outlet, the power supply is not a concern. The sizes and quantities of subdevice traffic packets after separation are shown in Table 3:

After preprocessing the data, we will show how to identify the types and statuses of ZigBee-based smart home devices from the traffic in two stages. In Chapter 6, we describe how to extract the ZigBee network key from the device's initial traffic and decrypt encrypted traffic in the ZigBee network. In Chapter 7, we introduce the methods for identifying the types and states of smart devices using machine learning models.

6. Device Initialization Privacy Security Issues

After separating the ZigBee network data, we extract from the device its initial traffic. In this chapter, we analyze the process of device initialization and extract the network key from the traffic. By using network keys to decrypt device data traffic, we can analyze device traffic in plaintext, which means that the encryption measures of the ZigBee network do not work.

6.1. Device Initialization Process. The device initialization process is shown in Figure 6. When a user wants to add a new device to the ZigBee network, the end device will first continuously broadcast the beacon request on the communication channel until the coordinator returns a beacon packet. If the network is off, the coordinator will not accept the beacon request, and the new device is not allowed to join the ZigBee network. When the coordinator switches from the network-closed state to the network open state, it will broadcast a permit join request packet to notify all devices that the network is open. The coordinator accepts the beacon request packet and broadcasts the beacon packet, which contains the coordinator's short address and extended address (EPID: Extended PAN ID). The coordinator's short address is generally 0x0000. The new device will then send an association request to the coordinator. The request packet takes the new device's MAC address as the source address and the coordinator's short address as the destination address. After receiving the request, the coordinator returns an association return packet to the terminal device, which contains the PAN ID of the ZigBee network and the short address assigned to the end device. After completing the negotiation process, the coordinator will send a packet containing the NWK key to the end device, which is encrypted by the global link key. The end device decrypts the packet using the global link key hardcoded in the firmware to obtain the NWK key. Finally, the end device successfully joins the ZigBee network, and then all the communication traffic at the network layer and above is encrypted using the NWK key. At the same time, the new device will broadcast its own short address and extended address to the network.

6.2. Network Key Extraction. The NWK key is present in the device initialization traffic data. The packet transmitting the key is an APS command followed by the association response packet. The source address of the packet is the coordinator's short address 0x0000, and the destination address is the new device's short address. The packet containing the NWK key can be easily extracted by analyzing the initial traffic.

The APS command frame is shown in Figure 7. The APS payload is encrypted with the global link key. This key is hardcoded in the firmware of both the coordinator and the subdevices. We can obtain the global link key of the device by extracting the device firmware and analyzing the content. In our experiment, it was found that all devices used the following default global link key: ZigBeeAlliance09 (5A 69 67 42 65 65 41 6C 6C 69 61 6E 63 65 30 39). This is an insecure setting that greatly reduces the difficulty for attackers to obtain the NWK key. An attacker must only decrypt the payload with the default global link key.

Wireshark [26] is a network protocol analyzer that can analyze ZigBee traffic. We entered the obtained initial traffic into Wireshark for analysis and input the default global link key in the settings. Wireshark can parse the content of the encrypted data and load the NWK key automatically. The NWK key parsed by Wireshark is shown in Figure 8. After that, we directly analyze the ZigBee plaintext data and obtain the activity information of the smart device.

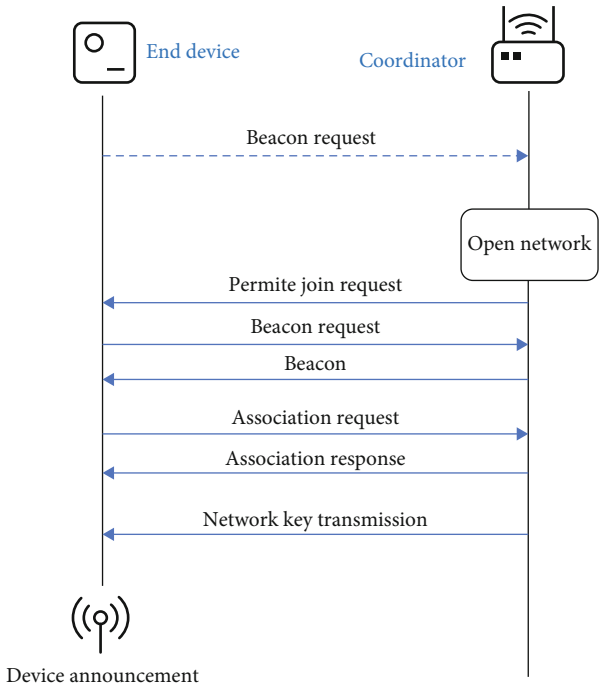

Figure 6: New end device network access process.

7. Smart Home Device Recognition

In practical scenarios, it is not easy to capture the initialization traffic of the device. When a user uses a smart device on a daily basis, once it is connected to the smart gateway, it will not need to be reconnected again unless the device malfunctions. In this section, we introduce a method for smart home device identification without the initialization traffic. We extract device traffic features through a sliding window and use a machine learning classification model to train the data. Even if the device traffic is encrypted, we can identify the types and statuses of ZigBee smart devices.

7.1. Smart Home Device Type Recognition. To breach the security and privacy of smart home devices, attackers need to first identify the types of devices present in the smart home. We process each traffic packet into the following structure:

$$\mathrm{pk}_i = (\mathrm{time}, \mathrm{len}, \mathrm{type}). \tag{1}$$

pk_i represents the i-th data packet collected, time represents the timestamp of the data packet, len represents the length of the data packet, and type represents the type of data packet. In the packet type, we define the packet sent by the smart gateway to other subdevices as 0, the packet sent by the subdevice to other devices as 1, the broadcast packet as 2, and the acknowledgment packet and other packet types as 3. After preprocessing ZigBee traffic packets, each device has a corresponding three-column data matrix. We use this matrix to extract the characteristics of different device types.

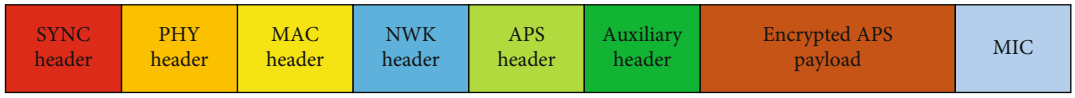

Figure 7: APS command frame.

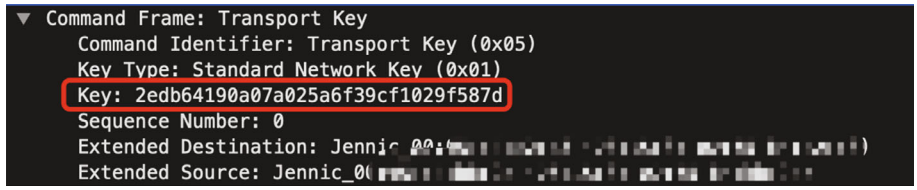

Figure 8: The NWK key.

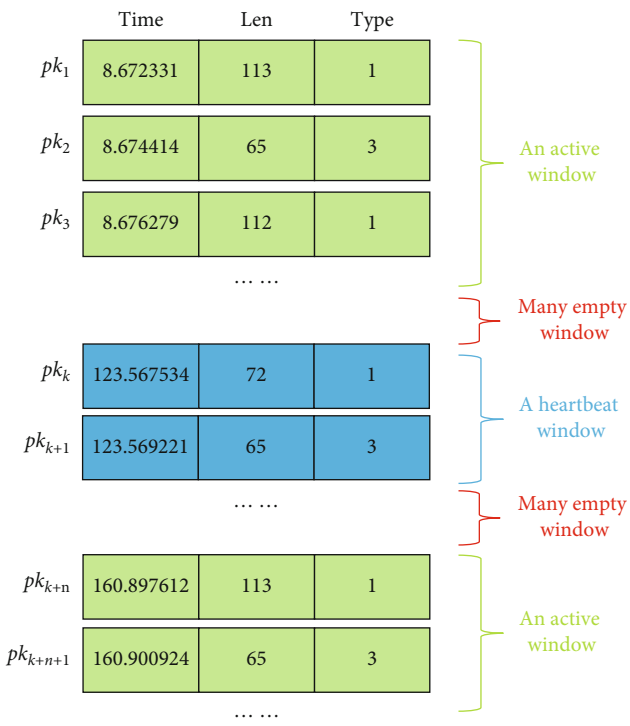

Figure 9: Time window example.

Through experimental observation, it is found that the complete process of triggering a smart device includes the following three steps: the user triggers the smart device, the smart device sends a data packet to the gateway, and the gateway returns an acknowledgment packet to the smart device. It only takes approximately a few seconds to complete the whole process. We regard the data traffic packet as a continuous process and set the time window to 5 seconds to include a smart device interaction process that is as complete as possible. We define the window containing the active event packet as the active window. When the device is not triggered, it will not generate trigger data packets, but it will send heartbeat packets with the data request command to the smart gateway at regular intervals to prove its online status. We define the window without data packets as the empty window and define the window containing only heartbeat packets as the heartbeat window. The time window of the device is shown in Figure 9. We calculate the traffic characteristics of the device every 5 seconds and use this to represent a sample of the characteristics of a device type.

We use device type features to represent each time window and combine each successive time window to generate a multicolumn matrix of feature sets. The feature matrix is stored in a file and used for subsequent classification model training. We divide the features of each time window into the following three categories: statistical features of trigger packets, statistical features of heartbeat packets, and device hard-coded features.

Device trigger packets are a major part of the overall ZigBee network traffic. Through the analysis of traffic data generated by different devices, it is found that there are differences in the number of traffic packets generated by some different types of devices in the interaction process, and there are some differences in the sizes of trigger packets. Therefore, for each time window, we calculate the average packet length and the standard deviation of the packet length, the number of packets, and the average return time as a sample of the characteristics of a device type.

Regarding the heartbeat packets, we found some differences in the idle periods of smart devices in the experiments. We count the number of empty windows between two heartbeat windows in idle periods and add the number to the feature set of the heartbeat window. For the active window, this feature is defined as 0.

In addition, there are some hard-coded device features. We found that the working channel of the ZigBee smart device will not change because the channel has been written into the device firmware during the production process. Since the working channels of different smart devices may be different, we also take the working channels of smart devices as a classification feature.

Singhin et al. [23] constructed a MAC address library and used the address database to distinguish different device types. The MAC address is the general ID assigned to each network device, and the MAC address of each device is different. A conventional MAC address is composed of six bytes. However, the ZigBee protocol is built based on 802.15.4, and the MAC address of each device consists of 8 bytes to provide more space for address allocation. IEEE provides the allocation data of the Organizationally Unique

Identifier (OUI) [27], including the MAC address prefix information, organization name, company address, country, and other information. Many MAC address identification databases have been built [28, 29], but no MAC address identification database for the 802.15.4 protocol has been found. Collecting a large number of ZigBee device MAC addresses is still a great challenge. In the ZigBee network, MAC address information is always transmitted in plaintext and is easy to obtain. The first three bytes of the device's MAC address can be converted into the corresponding manufacturer or supplier information. We generalize different numbers for different manufacturers as a new device feature.

The device type feature set is shown in Table 4.

The device type features are entered into the machine learning classification model to train the model after the device traffic feature extraction. To solve the imbalance of feature data between devices, we reduced the data on some devices that had too many packets, especially those continuously powered by sockets, such as Xiaomi's natural gas alarm. Finally, we obtain the specific type of smart devices from the classification model.

7.2. Smart Home Device Status Recognition. After completing the device type identification of smart home devices, the specific device types are obtained. In this part of the work, we will further explore the problem of smart home privacy leakage and identify the specific state of the device. State recognition of devices can lead to leakage of user privacy in smart homes. For example, an attacker can determine whether a human body sensor is moving and thus obtain information on the activities of people in the smart home can be obtained.

Some operation modes of the same device in different states are quite different, and some operation modes are slightly different. We analyzed the smart home devices of different manufacturers and found that the specific states of some devices cannot be identify. In this paper, the experimental devices are divided into the following three categories:

(1) Class I: Haiman body sensor, Jingdong wireless switch, Jingdong body sensor, Xiaomi body sensor, Aqara vibration sensor, Aqara body sensor, and Huawei body sensor

(2) Class II: Haiman door sensor, Jingdong door sensor, Jingdong wireless smart socket, Aqara door sensor, and Huawei door sensor

(3) Class III: Huawei temperature and humidity sensor, Xiaomi smoke alarm, and Xiaomi gas alarm

The specific state types of class I devices can be identified. There are differences in the operating modes of these devices in different states. We can build a state recognition model to identify the specific states for class I devices.

The specific statuses of class II devices cannot be recognized. This type of device has the following two states: on and off. From the perspective of plaintext, the data packets in the on state and off states differ by only one byte, and their interaction logic and data packet length are the same. After data packets are encrypted, the specific device status cannot be identified by using only the traffic analysis method. However, we can identify state change events for these devices. To a certain extent, changes in device state can also reveal the privacy activity of the smart home. For example, we can judge that someone in the smart home has opened or closed the doors by a change in the state of the door sensor.

Table 4: Device type feature set.

Category	Feature
Trigger packets	Mean packet length
	Packet number
	Standard deviation in packet lengths
	Mean interarrival time
Heartbeat packets	Empty window number
Hard-coded characters	MAC address
	Channel ID

Class III devices do not participate in smart home status recognition. On the one hand, some devices only monitor information in the natural environment and are not affected by human activities, such as Huawei's temperature and humidity sensors. However, some devices have harsh trigger conditions. We did not collect the data flow under the trigger mode of these devices, such as the Xiaomi smoke alarm and Xiaomi natural gas alarm.

To facilitate the extraction of the device state features, each packet is processed into the following structure:

$$\mathrm{pk}_i = (\mathrm{time}, \mathrm{len}, \mathrm{type}, \mathrm{devstate}). \tag{2}$$

Assuming that the state set of a device is θ, each device state is $a_i \in \theta$, and a device owns state $\{a_1, a_2, \cdots\} \in \theta$. There are a large number of data packets within the duration of each state. The data packet sequence $S_i^{a_i} = (pk_1, pk_2, pk_3, \cdots, pk_n)$ represents the set of data packets of the device in state a_i and is used as a sample to construct the complete dataset of the device state. Since smart devices constantly switch their states due to user behavior, the complete traffic of a smart device is constructed as a sequence $T_{\mathrm{device}} = (S_1^{a_1}, S_1^{a_2}, S_2^{a_1}, \cdots, S_i^{a_i})$.

In Chapter 7.1, we obtained the specific device type. We only need to establish a state classification model for each smart device instead of mixing all device states. This measure helps us improve the device state recognition accuracy. The training data of the classification model require the state label of the device, but the data packets containing the device state information are encrypted with the network key. To label the device data packets accordingly, we need to capture the device traffic when the device enters the network and obtain the network key of the ZigBee network according to the method in Chapter 6. After that, we can decrypt the traffic data of the device and obtain the status information of the device. The construction of the state sample set of the smart device is shown in Figure 10.

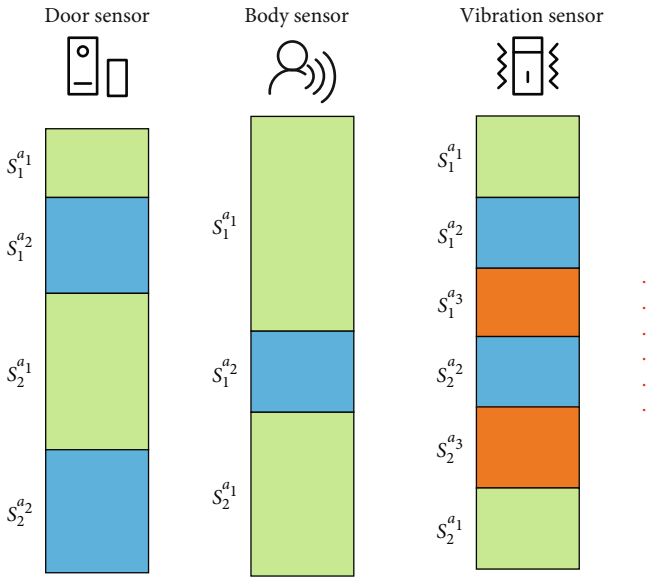

Figure 10: Construction of the state sample set.

To extract the features of the device in different states, we extract the relevant features from the time series to identify the specific state. We use the tsfresh [30] tool to automatically calculate a large number of time series features and construct the feature vectors of different states for each device. The tsfresh tool is used to extract custom state features from the state sample set, such as the absolute energy value of the time series, first-order difference total sum, Fourier transform coefficient, and approximate entropy. Because the tsfresh tool will generate many state features that are not all related to the device state, some redundant features will mislead the device state recognition model and lead to more time spent on model training. Reducing irrelevant features prevents model overfitting, optimizes model performance, and improves the accuracy of device state identification. The number of optimized features of each device is shown in Table 5.

8. Experimental Evaluation and Analysis

To study the privacy and security of smart homes based on ZigBee, we collected 14.49 MB of traffic from 20 different devices from 5 manufacturers for the experiment. We first explore how to identify the specific type of device in a smart home environment based on ZigBee encrypted traffic. In the experiment, the 5-fold cross-validation method is used to evaluate the model's performance to verify the model's effectiveness. The 5-fold cross-validation method divides the training dataset into five parts, takes four parts as the training dataset, one part for performance evaluation, and then repeats it five times to avoid problems caused by unreasonable dataset division. The average value of five experiments represents the model's performance. Nine common machine learning classification models are used to train the dataset during the experiment.

The experimental data for device type identification are divided into two parts. The first involves dividing the data according to different manufacturers to explore the accuracy of identifying each device from the same manufacturer. The second is to integrate the data of various manufacturers to explore the model's ability to identify multiple manufacturers and multiple devices.

Table 5: Feature numbers for the devices.

Smart device	Feature number
Jingdong-wireless smart socket	214
Jingdong-body sensor	32
Jingdong-wireless switch	15
Jingdong-door sensor	28
Xiaomi-body sensor	85
Aqara-door sensor	103
Aqara-vibration sensor	141
Aqara-body sensor	500
Haiman-body sensor	385
Haiman-door sensor	315
Huawei-body sensor	185
Huawei-door sensor	162

The device type identification experiment is divided into two stages. First of all, the statistical features of trigger packets are used as essential features and entered into the classification model. The characteristic matrix is a 5-column array. The experimental results are shown in Table 6.

The experimental results show that the RandomForest, DecisionTree, ExtraTree, and XGBoost models can achieve better overall accuracy, i.e., approximately 85% accuracy. The XGBoost model has the highest device recognition rate (85.32%). When classifying single-manufacturer devices, the other classification models still perform acceptably. However, when all device data are combined, the performance levels of these classification models drop significantly, and the recognition accuracy levels are much lower than the performance levels of the above four models. For different manufacturers, we found that different models can achieve high performance when classifying internal devices from Xiaomi and Apara. For example, the XGBoost model has an accuracy rate of 96.13% when classifying Xiaomi devices and 98.48% when classifying Apara devices. However, the ability to classify the Jingdong, Haiman, and Huawei devices is not as good, and most of the accuracy rates range from 80% to 83%.

To improve the type classification accuracy of the overall equipment, we added the statistical features of heartbeat packets and device hard-coded features into the feature vector. The characteristic matrix is a 7-column array. We retrained the device classification model with the above machine learning classification model. The experimental results are shown in Table 7.

From the experimental results, new features can effectively improve the overall device recognition rate. The random forest classification model obtains the highest device recognition rate (93.27%). Compared with the previous experiment, the highest device recognition rate is improved by nearly 8%. The LSTM model also achieved relatively high

Table 6: ZPA-1: The device type identification model performance with base features.

Classifier	Jingdong (%)	Xiaomi (%)	Haiman (%)	Apara (%)	Huawei (%)	Total (%)
RandomForest	86.62	96.02	80.55	98.04	83.56	85.19
DecisionTree	86.49	96.06	80.54	98.37	83.70	85.30
ExtraTree	86.62	96.13	80.06	98.48	83.49	85.24
SVM	74.16	89.84	72.87	72.69	75.12	53.31
Logistic Reg.	84.46	95.69	78.59	80.93	74.61	62.94
Naïve Bayes	75.21	84.54	70.09	72.69	71.12	50.65
KNN	85.26	95.58	78.59	97.50	81.52	81.83
XGBoost	86.74	96.13	80.38	98.48	83.49	85.32
Adaboost	86.43	95.91	79.73	98.04	83.34	81.30
CNN	83.63	95.04	78.36	81.73	76.06	83.55
LSTM	72.21	93.82	80.45	71.82	76.18	84.06

Table 7: ZPA-2: the device type identification model performance with addition features.

Classifier	Jingdong (%)	Xiaomi (%)	Haiman (%)	Apara (%)	Huawei (%)	Total (%)
RandomForest	89.95	99.72	79.83	98.16	83.49	93.27
DecisionTree	89.71	99.74	80.48	98.47	83.63	93.16
ExtraTree	89.81	99.74	80.97	98.01	83.70	93.13
SVM	73.93	99.53	73.01	71.77	75.12	82.33
Logistic Reg.	78.37	99.34	77.39	77.34	74.61	85.46
Naïve Bayes	69.78	89.93	71.38	71.01	71.92	70.42
KNN	87.16	99.74	78.69	96.64	81.52	91.98
XGBoost	89.78	99.72	80.81	98.32	83.56	92.32
Adaboost	89.81	99.74	80.65	98.55	83.20	88.91
CNN	84.41	99.66	80.49	84.41	77.09	87.23
LSTM	75.34	99.63	82.92	71.10	77.81	91.92

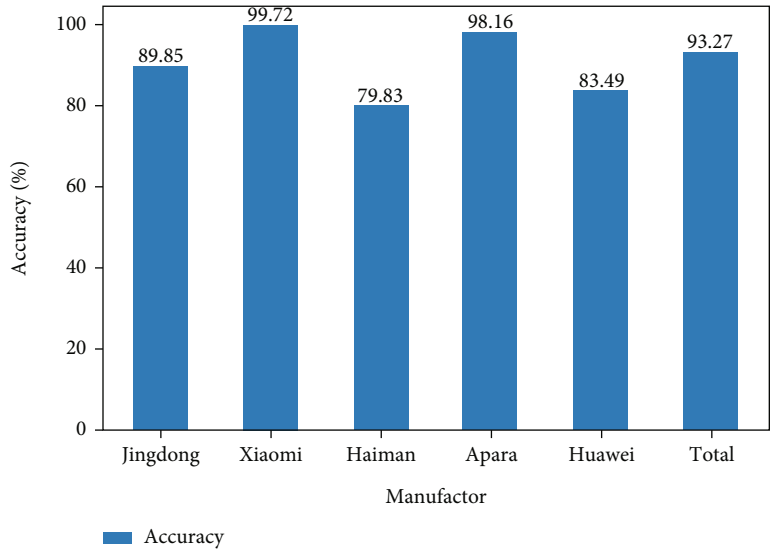

Figure 11: RandomForest model device type recognition performance.

accuracy, but training the model requires considerable time and memory. By comparing the device identification performance of each model, the RandomForest model is selected as the final device type identification model. For the RandomForest model, assuming that n is the number of training samples, f is the number of features, k is the number of trees, p is the number of nodes in the tree, and d is the depth of the tree, then the training time complexity of the model is $O(n$

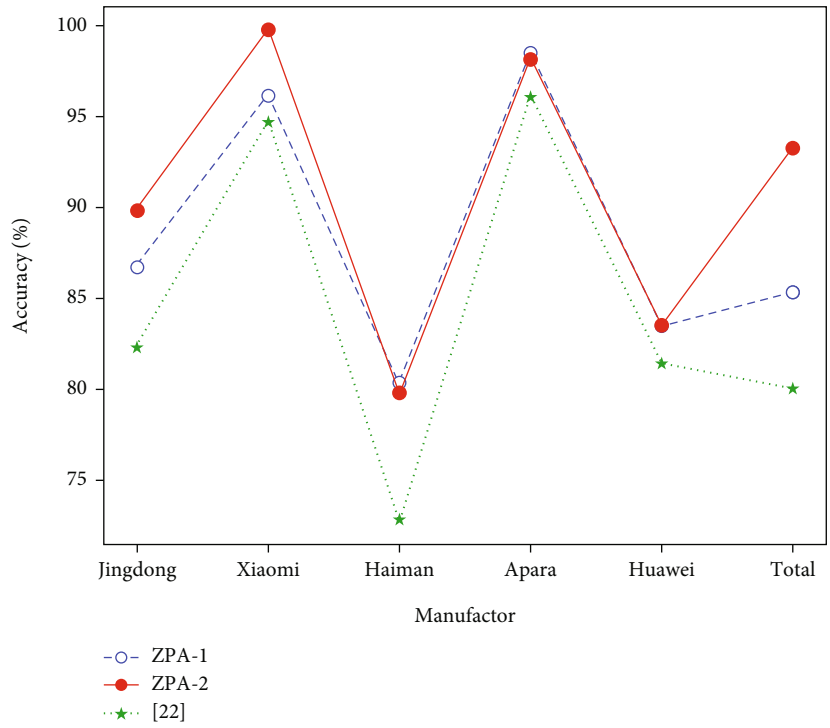

Figure 12: Device type identification performance of each method model.

$* \log(n) * f * k)$, the prediction time complexity is $O(d * k)$, and the runtime space complexity is $O(p * k)$. The results are shown in Figure 11.

To verify the superiority of our method, we compare the two experimental methods of device type identification with that of Acar et al. [22], which is also based on ZigBee traffic. ZPA-1 represents the method that uses essential features, and ZPA-2 represents that with newly added features. The result is shown in Figure 12, indicating that ZPA-2 has higher accuracy in the type recognition of multiple devices.

After identifying the specific smart device type, we further identify the state of the device based on its type. Through the tsfresh tool, we obtained the status-related characteristics of each smart device. To verify the performance results of different classification models in device state recognition and select the best classification model, some data are first used for model training. Similarly, the 5-fold cross-validation method is used to evaluate the performance levels of classification models. The results are shown in Table 8.

From the above experimental results, the decision tree has the best performance in terms of the device state recognition accuracy, reaching 94.92% under 5-fold cross-validation. Therefore, we use device state data to train the DecisionTree model to verify the effectiveness of device state recognition. For the DecisionTree model, assuming n is the number of training samples, f is the number of features, p is the number of nodes in the tree, and d is the depth of the tree, the training time complexity of the model is $O(n * \log(n) * f)$, the prediction time complexity is $O(d)$, and the runtime space complexity is $O(p)$. The experiment will identify the specific device state of the CLASS I devices and the state change of CLASS II devices. Each device has a feature matrix containing a different number of features and uses it to build the corresponding decision tree model. The experimental results are shown in Table 9.

Table 8: Device state identification model evaluation.

Classifier	5-fold cross-validation (%)
RandomForest	93.79
DecisionTree	94.92
ExtraTree	92.63
SVM	57.04
Logistic Reg.	92.65
Naïve Bayes	62.04
KNN	88.65
XGBoost	93.36
Adaboost	93.21
CNN	89.64
LSTM	90.52

The results show that the average smart device state recognition rate is 98.65%, while the status recognition rates of some devices reached 100%, and the lowest device status recognition rate is 93.77%. The result shows that even if the ZigBee traffic is encrypted, we can distinguish the specific state of the device accurately. At the same time, the experiment also reveals the current privacy leakage problem of smart home devices based on ZigBee. When the attacker is able to identify the state of each device in the smart home, he can construct the user's behavior pattern and analyze the user's daily behavior, which further threatens the user's personal and property safety.

Table 9: Smart device state recognition model performance.

Smart device	State	5-fold cross-validation (%)
Jingdong-wireless smart socket	Status change (on/off)	97.66
Jingdong-body sensor	Someone passed No one passed	100
Jingdong-wireless switch	Click	100
Jingdong-door sensor	Status change (on/off)	100
Xiaomi-body sensor	Someone passed No one passed	99.80
Aqara-door sensor	Click	95.24
Aqara-vibration sensor	Shock Drop Tilt	93.77
Aqara-body sensor	Someone passed No one passed	100
Haiman-body sensor	Someone passed No one passed	99.33
Haiman-door sensor	Status change (on/off)	99.45
Huawei-body sensor	Someone passed No one passed	99.88
Huawei-door sensor	Status change (on/off)	98.85
Average		98.65

9. Defensive Discussion

9.1. Defensive Measures. To address the security challenges faced by smart home devices, many researchers have performed much research work to ensure the security of smart home devices. Alrawi et al. [31] proposed a method to analyze the security of home IoT devices and systematized the existing smart home literature into attack technology, mitigation measures, and stakeholders. Aiming at the high false-positive rate of data mining technology, Fu et al. [18] proposed a semantic-aware anomaly detection system suitable for smart homes. The system can extract semantics from applications and their configuration files, mine user behaviors and device correlation between physical events, and monitor for abnormal states of intelligent devices. Ding et al. [17] proposed a novel dynamic security policy enforcement system called IOTSAFE. They utilize static analysis and dynamic testing techniques to capture the physical interactions between devices in a smart home environment, predict potentially dangerous situations, and block unsafe device states. Zhang et al. [16] designed and developed a HoMonit system using the side-channel inference function. HoMonit inferred the SmartApps activities from encrypted traffic, compared them with the expected behavior in the source code or UI interface, and monitored the abnormal behavior of SmartApps. Wang et al. [32] analyzed the attack landscape of ZigBee-enabled IoT systems and proposed a new certificate-less ZigBee-joining protocol that leverages low-cost public-key primitives. Fan et al. [33] analyzed the security policies, measures, and architecture of the ZigBee protocol and discussed some devices and methods used to find security vulnerabilities. There are also some studies [34, 35] that designed new secure cryptographic algorithms for low-power IoT devices.

In the attack scenario studied in this paper, the above defense measures cannot prevent traffic analysis attacks. Reducing the attacker's access to user privacy through the air traffic of smart home devices is still a complex problem. In the work of Acar et al. [22], a solution based on generating forged traffic is proposed, which uses fake traffic to mask real user activity. However, the authors directly study the impact of false packet injection on model performance by modifying the eigenvectors without considering the feasibility in real environments. This study also fails to consider whether this forged traffic will affect the regular operation of the device or whether the attacker can identify the false traffic. It is feasible to restrict attackers from obtaining private user data through traffic analysis, generating false traffic, and establishing a confrontation model. It is necessary to consider who generates the virtual data. In this regard, an independent third-party protective device is more suitable for practical use. If the smart device generates false traffic, it will increase power consumption and reduce the user experience. In addition, we also need to consider enhancing the authenticity of false traffic to prevent attackers from identifying false traffic by retraining the model.

9.2. Limitations and Future Work. Since the ZigBee device identification model relies on ZigBee traffic captured over the air, there must be a corresponding sniffer device around the smart home devices. Our system can detect privacy only over a short distance and cannot be directly used for the remote analysis of user privacy. Smart home devices can communicate with cloud services through Access Points (APs), but we do not use this part of the traffic. When the attacker has control of the AP or even the Internet Service Provider (ISP), there will be a greater advantage than that obtained through a local attack, and we will study this in the future. In the meantime, using eXplainable Artificial Intelligence (XAI) to provide global interpretation [36] is also an interesting direction.

Some ZigBee smart home devices can be controlled remotely via apps, such as wireless switches. Even if our system recognizes the corresponding smart home device event, it does not mean that the user triggers the corresponding device locally, as the user may control the smart home device through the cloud, which gives the attacker the illusion that someone is active at home. Our model cannot tell whether the user is triggering the device locally or controlling the device remotely. We will study this issue in the future. In addition, by obtaining the specific activities of the device, we can further use this information to construct a user

activity model, divide the user's family into multiple areas, and further refine the user's activities. This work will be very interesting. In the real environment, there are single-user households and multiuser households, and it is very challenging to infer user activities from a complex environment.

10. Conclusions

This paper proposes a smart home privacy analysis system called ZPA based on ZigBee encrypted traffic and reveals the privacy and security issues of ZigBee-based smart homes. By passively capturing air ZigBee communication packets, we can extract the network key from the initial traffic of the device and then extract the operating characteristics of the device to train the machine learning classification model. Through our experiments, we found an unsafe configuration problem in ZigBee devices. Our method can effectively improve the type recognition accuracy of multiple similar devices produced by multiple manufacturers. Even if device traffic is encrypted, we can identify the type of ZigBee smart device and infer its state. ZigBee-based smart home devices still have shortcomings in terms of privacy protection. In the future, further research on effective wireless signal privacy protection methods is needed.

Abbreviations

ZPA: ZigBee protocol analysis system
IoT: Internet of things
IEEE: Institute of electrical and electronics engineers
PHY: Physical layer
MAC: Media access control layer
NWK: Network layer
APL: Application layer
APS: Application support sublayer
ZDO: ZigBee device object
TC: Trust center
VPN: Virtual private network
NAPT: Network address port translation
LSTM: Long short-term memory
NIDS: Network intrusion detection system
AES: Advanced Encryption Standard
PAN ID: Personal area network ID
BLE: Bluetooth low energy
WPA: Wi-Fi protected access
TLS: Transport layer security
EPID: Extended PAN ID
OUI: Organizationally unique identifier
AP: Access point
ISP: Internet service provider
XAI: eXplainable artificial intelligence.

Data Availability

The experimental data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The author declares that there are no conflicts of interest.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (2019YFB2101700).

References

[1] MiSecurity, "Cyber security baseline for consumer internet of things device," Feb. 2022, https://github.com/MiSecurity/Cyber-Security-Baseline-for-Consumer-Internet-of-Things/blob/main/resources/pdf/Cyber.

[2] Z. Alliance, "Zigbee specification," Apr. 2022, https://csa-iot.org/all-solutions/zigbee/.

[3] C. L. Corbett, R. A. Beyah, and J. A. Copeland, "Passive classification of wireless NICs during active scanning," *International Journal of Information Security*, vol. 7, no. 5, pp. 335–348, 2008.

[4] S. Bratus, C. Cornelius, D. Kotz, and D. Peebles, "Active behavioral fingerprinting of wireless devices," in *Proceedings of the 1st ACM Conference on Wireless Network Security*, pp. 56–61, Alexandria VA USA, 2008.

[5] K. Gao, C. Corbett, and R. Beyah, "A passive approach to wireless device fingerprinting," in *Proceedings of the International Conference on Dependable Systems and Networks*, pp. 383–392, Chicago, IL, 2010.

[6] C. L. Corbett, R. A. Beyah, and J. A. Copeland, "Using active scanning to identify wireless NICs," in *Proceedings of the 2006 IEEE Workshop on Information Assurance*, pp. 239–246, West Point, NY, USA, 2006.

[7] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J. van Randwyk, and D. Sicker, "Passive data link layer 802.11 wireless device driver fingerprinting," *USENIX Security Symposium*, vol. 3, pp. 16–89, 2006.

[8] D. C. C. Loh, C. Y. Cho, C. P. Tan, and R. S. Lee, "Identifying unique devices through wireless fingerprinting," in *Proceedings of the 1st ACM Conference on Wireless Network Security*, pp. 46–55, Alexandria VA USA, 2008.

[9] S. Jana and S. K. Kasera, "On fast and accurate detection of unauthorized wireless access points using clock skews," *IEEE Transactions on Mobile Computing*, vol. 9, no. 3, pp. 449–462, 2010.

[10] C. Neumann, O. Heen, and S. Onno, "An empirical study of passive 802.11 device fingerprinting," in *Proceedings -32nd IEEE International Conference on Distributed Computing Systems Workshops*, pp. 593–602, Macau, China, 2012.

[11] A. K. Dalai and S. K. Jena, "WDTF: a technique for wireless device type fingerprinting," *Wireless Personal Communications*, vol. 97, no. 2, pp. 1911–1928, 2017.

[12] M. R. Shahid, G. Blanc, Z. Zhang, and H. Debar, "IoT devices recognition through network traffic analysis," in *2018 IEEE International Conference on Big Data*, pp. 5187–5192, Seattle, WA, USA, 2018.

[13] B. Bezawada, M. Bachani, J. Peterson, H. Shirazi, I. Ray, and I. Ray, "Behavioral fingerprinting of internet-of-things devices," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 11, no. 1, article e1337, 2021.

[14] S. Dong, Z. Li, D. Tang, J. Chen, M. Sun, and K. Zhang, "Your smart home can't keep a secret: towards automated fingerprinting of IoT traffic," in *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, pp. 47–59, Taipei Taiwan, 2020.

[15] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. R. Sadeghi, and S. Tarkoma, "Iot sentinel: automated device-type identification for security enforcement in iot," in *Proceedings - International Conference on Distributed Computing Systems*, pp. 2177–2184, Atlanta, GA, USA, 2017.

[16] W. Zhang, Y. Meng, Y. Liu, X. Zhang, Y. Zhang, and H. Zhu, "HoMonit-monitoring smart home apps from encrypted traffic," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1074–1088, Toronto Canada, 2018.

[17] W. Ding, H. Hu, and L. Cheng, "IOTSAFE: enforcing safety and security policy with real IoT physical interaction discovery," in *the 28th Network and Distributed System Security Symposium (NDSS 2021)*, 2021.

[18] C. Fu, Q. Zeng, and X. Du, "HAWatcher: semantics-aware anomaly detection for appified smart homes," in *30th USENIX Security Symposium (USENIX Security 21)*, pp. 4223–4240, 2021.

[19] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," 2018, https://arxiv.org/abs/1802.09089.

[20] G. Bovenzi, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescape, "A hierarchical hybrid intrusion detection approach in IoT scenarios," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*, pp. 1–7, Taipei, Taiwan, 2020.

[21] N. Apthorpe, D. Reisman, and N. Feamster, "A smart home is no castle: privacy vulnerabilities of encrypted iot traffic," 2021, http://arxiv.org/abs/1705.06805.

[22] A. Acar, H. Fereidooni, T. Abera et al., "Peek-a-boo: I see your smart home activities, even encrypted," in *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pp. 207–218, Linz, Austria, 2020.

[23] A. D. Singh, L. Garcia, J. Noor, and M. Srivastava, "I always feel like somebody's sensing me! a framework to detect, identify, and localize clandestine wireless sensors," in *30th USENIX Security Symposium (USENIX Security 21)*, pp. 1829–1846, 2021, July 2021. https://www.usenix.org/system/files/sec21fall-singh.pdf.

[24] Y. Cheng, X. Ji, T. Lu, and W. Xu, "DeWiCam: detecting hidden wireless cameras via smartphones," in *Proceedings of the 2018 ACM Asia Conference on Computer and Communications Security*, pp. 1–13, Incheon Republic of Korea, 2018.

[25] Secdev, "Scapy," May 2021, https://github.com/secdev/scapy.

[26] Wireshark Foundation, "The wireshark network protocol analyzer," 2022, https://code.wireshark.org/review/gitweb?p=wireshark.git.

[27] IEEE, "Oui," 2021, June 2021, http://standards-oui.ieee.org/oui/oui.txt.

[28] J. Bauer, "mac_vendor_lookup," 2020, June 2021, https://github.com/bauerj/mac_vendor_lookup.

[29] Pwnie Express, "Louis," 2019, June 2021, https://github.com/pwnieexpress/louis.

[30] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, "Time series feature extraction on basis of scalable hypothesis tests (tsfresh - a python package)," *Neurocomputing*, vol. 307, pp. 72–77, 2018.

[31] O. Alrawi, C. Lever, M. Antonakakis, and F. Monrose, "SoK: security evaluation of home-based IoT deployments," in *Proceedings - IEEE Symposium on Security and Privacy*, pp. 1362–1380, San Francisco, CA, USA, 2019.

[32] W. Wang, F. Cicala, S. R. Hussain, E. Bertino, and N. Li, "Analyzing the attack landscape of Zigbee-enabled IoT systems and reinstating users' privacy," in *WiSec 2020 - Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pp. 133–143, Linz Austria, 2020.

[33] X. Fan, F. Susan, W. Long, and S. Li, *Security Analysis of Zigbee*, MWR InfoSecurity, 2017, http://ieeexplore.ieee.org/document/7940247/%0Ahttps://courses.csail.mit.edu/6.857/2017/project/17.pdf%0Ahttps://www.mwrinfosecurity.com/assets/Whitepapers/mwri-zigbee-overview-finalv2.pdf.

[34] S. Prajapat, D. Rajput, and R. S. Thakur, "Time variant approach towards symmetric key," in *Proceedings of 2013 Science and Information Conference*, pp. 398–405, London, UK, 2013.

[35] S. Prajapat, S. Swami, B. Singroli, R. S. Thakur, A. Sharma, and D. Rajput, "Sparse approach for realizing AVK for symmetric key encryption," *International Journal of Recent Development in Engineering and Technology*, vol. 2, no. 4, pp. 15–18, 2014.

[36] A. Nascita, A. Montieri, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescape, "XAI meets mobile traffic classification: understanding and improving multimodal deep learning architectures," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4225–4246, 2021.