Identifying Internet of Things Devices Using Machine Learning Based on Encrypted Traffic Data

Seung-Ju Han, Dong-Hyuk Shin, Yu-Bin Kim,
Jong-Bum Lee, and Ieck-Chae Euom*

Chonnam National University
{Sjhan, shindh2, kingyoubin97, zmdk154112, iceuom}@jnu.ac.kr

Abstract

IoT technology's rapid evolution and enhanced connectivity have enabled its widespread application from infrastructural settings to everyday households as smart homes. With the increasing prevalence of IoT devices, concerns about data protection have grown, leading to the adoption of security measures like encryption in common lightweight communication protocols. However, using these encryption techniques can hinder the identification of devices participating in communication and complicate network traffic analysis during intrusion investigations. In this study, therefore, we extracted features from encrypted traffic data and conducted IoT device fingerprinting through classification algorithms. Experimental results revealed that the Random Forest classification algorithm achieved a maximum classification accuracy of about 93.89%.

Keywords: Machine Learning, IoT, Encrypted Traffic, Device Identification

1 Introduction

With the progression of the fourth industrial revolution, there is an increased interconnectivity among Internet of Things (IoT) devices. These devices now facilitate various services across various domains, enriching user experiences. For instance, industrial facilities, such as factories, are serviced with IoT under the moniker of Industrial Internet of Things (IIoT). Concurrently, IoT services catering to home automation are branded as Smart Home.

The 7th International Conference on Mobile Internet Security (MobiSec'23), Dec. 19-21, 2023, Okinawa, Japan, Article No. 27
*Corresponding author: Chonnam National University, Daejeon, 34134, Republic of Korea

However, the soaring demand for IoT has sufficiently piqued the interest of cyber attackers. Since the advent of the Mirai botnet in 2016, targeting IoT services, there has been a consistent emergence of various malicious codes based on it. These codes often disrupt the regular functionalities of IoT-

based services. Moreover, beyond these malicious codes, some incidents involve exploiting vulnerabilities in IoT devices leading to specific intrusions.

Although cyberattacks targeting IoT services are on the rise, standardized procedures for IoT digital forensics remain absent. As a foundation for comprehensive digital forensics, professionals often resort to the NIST SP 800-86 document[1] to investigate intrusions concerning IoT. NIST SP 800-86 delineates the initial phase of digital forensics as the identification of data sources followed by data collection. It underscores the necessity to assess, extract, and analyze traffic data related to network activities, ensuring an understanding of the implications on systems and networks.

However, identifying IoT devices poses challenges due to various reasons. IoT devices exhibit heterogeneous characteristics depending on their intended usage – varying in form, structure, and the network protocols they employ. Such heterogeneity signifies the difficulty in applying a standardized method or framework for device identification. Additionally, the evolution of encryption algorithms to counter cyberattacks can further hinder device identification[2].

To facilitate efficient intrusion investigations, there is a need for methodologies capable of identifying IoT devices across arbitrary protocols without decrypting network packets. In this study, we employed machine learning-based classification algorithms to identify devices using encrypted network data and assessed the accuracy of identification.

2 Related Works

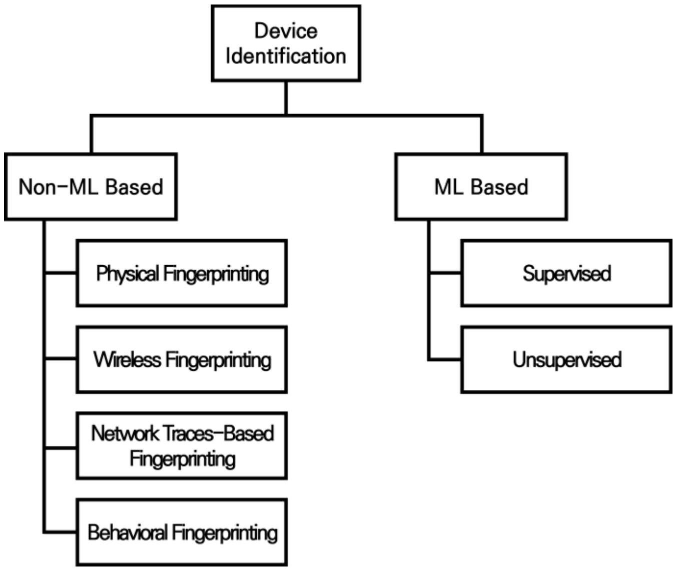

Figure 1: Classification of device Identification techniques

To assess the extent of asset damage and analyze events due to security breaches, it's crucial to identify the devices involved in the breach. Device identification technology can be broadly categorized into two types as shown in Figure 1: non-machine learning-based device identification and machine learning-based device identification. Non-machine learning-based device identification primarily employs sophisticated rules and algorithms, like fingerprinting. Device identification via fingerprinting typically adopts the most reliable result from a vast fingerprinting database filled with device information.

Machine learning-based device identification extracts meaningful features from device behavior information, such as network traffic, and determines the type of device based on the most reliable result from a trained model. Machine learning-based device identification can be largely classified into supervised and unsupervised learning. Supervised learning for device identification characteristically provides high accuracy for well-known devices with substantial data. However, caution is needed when dealing with devices with insufficient data or unknown devices, as they might not be identified correctly. In the case of unsupervised learning for device identification, it can classify unknown devices. Nevertheless, as precise label data for collected data isn't provided, there's a possibility of inaccurate results, warranting caution.

2.1 Device Identification Based on Non-Machine Learning

To analyze the scale of asset damage and events due to security intrusions, it's essential to identify the devices involved in the intrusion. Noman Mazhar et al. have argued that the identification of IoT devices is mainly carried out through classification and device fingerprinting[3]. To effectively identify malicious IoT devices, it is crucial to utilize both methods concurrently. They categorized the fingerprinting technologies into four types as illustrated in Figure 1: physical fingerprinting, wireless fingerprinting, network traces-based fingerprinting, and behavioral fingerprinting.

Physical Fingerprinting identifies a device based on its physical characteristics, rather than its logical features. It refers to the physical components of the device that can be treated as identifiers. Wireless Fingerprinting refers to the identification of devices through characteristics observed in wireless communication. For instance, the signal strength of Wi-Fi is often used to identify devices in this manner[4]. Network Traces-Based Fingerprinting identifies devices based on patterns in network activity. The primary goal is to understand a device's actions based on the metadata of network packets and to classify the device accordingly. Behavioral Fingerprinting identifies devices based on the unique behavioral patterns exhibited during network communication.

Among these four device fingerprinting techniques, both Network Traces-Based Fingerprinting and Behavioral Fingerprinting can be categorized as methods based on network traffic for device identification. However, since the primary network activity of IoT devices is recommended to be encrypted, it is challenging to accurately discern the behavior of the device. ISO/IEC 27400, which defines security guidelines for IoT devices, emphasizes the necessity of encrypting transmitted information regarding network and communication technologies applied to IoT and systems[5]. Moreover, lightweight wireless communication protocols predominantly used by IoT devices provide encryption techniques suitable for each protocol and the nature of the corresponding device. This makes it easier for IoT service developers and providers to encrypt information.

2.2 Device Identification Based on Machine Learning

Table 1: Comparison of related study

Reference	Target Protocol			
	Zigbee	TCP/UDP	Z-Wave	BLE.
[6]	O			
[7]	O	O		
[8]	O		O	
[9]				O

Table 1 summarizes a study conducted on device identification based on network behavior using supervised machine learning. Hossein Jafari et al.[6] conducted a study on the identification of Zigbee sensors using a sensor board that incorporated temperature, lighting, acceleration, magnetism, and voice sensors. By employing classification models such as DNN, CNN, and LSTM, they successfully classified the sensors with an accuracy of up to 96.3%. Ola Salman et al.[7] performed study in an experimental environment composed of sensors and actuators. They extracted header information from Zigbee and TCP/UDP communication packets to detect attacks and identify devices. Using the RF algorithm, they achieved a maximum accuracy of 97%. Leonardo Babun et al.[8] undertook a device identification experiment targeting IoT devices using Z-Wave and Zigbee protocols, demonstrating an identification accuracy of up to 93.25%. Jinghui Zhang et al.[9] explored a method for device identification by extracting features from the data link layer of BLE communication. The accuracy achieved in this study approached 99.8%.

These studies primarily focused on extracting meaningful information from encrypted packets by emphasizing network packet headers and displayed commendable accuracy rates. However, most of these investigations face challenges in evolving into comprehensive classifier studies due to the limited number of target protocols or device types. Extracting meaningful information from the payload due to the encrypted communication of Internet of Things (IoT) devices is an intricate task. Given that the type of information embedded in packets can significantly vary depending on the communication protocol, constructing a device identification model that is independent of the protocol is no simple feat. Thus, this study built a classification model by extracting commonly utilized information, which aids in device identification, from the unencrypted header section.

3 Proposal Model

Figure 2 depicts the diagrammatic representation of the proposed approach in this study for device identification in IoT using machine learning based on encrypted packet data. Initially, the packet data from the network, which is intended for device identification, is gathered. Subsequently, features used for device identification are extracted from the collected network packet data header information. These extracted features are transformed into the Flow Object proposed in this study, possessing 16 features to be utilized in the device identification machine learning model. Finally, based on the features of each Flow Object, a machine learning model is developed to identify which device participated in that particular network communication.

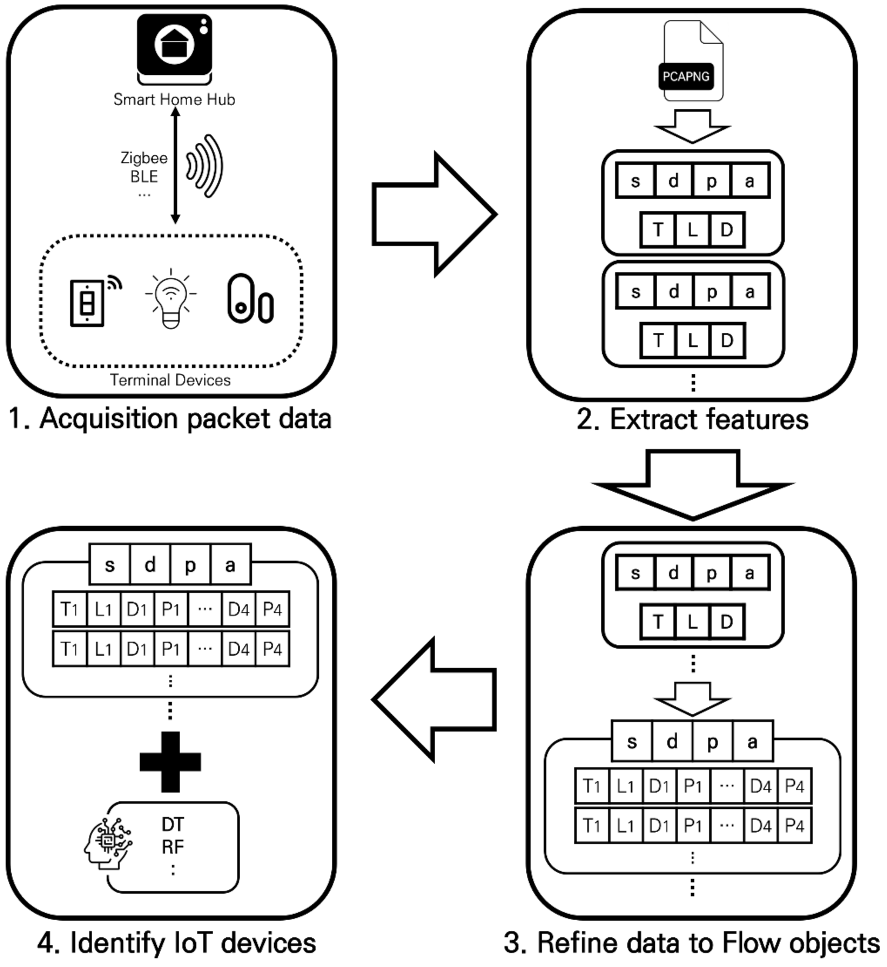

Figure 2: Diagram for proposed device identification

3.1 Collect Packet Data

The method of collecting packet data participating in the network varies depending on which protocol the surrounding IoT devices communicate with. For instance, for the Zigbee protocol, data can be collected using the CC2531 module, while for the BLE protocol, packet sniffing can be conducted using the Ubertooth-one module. By following the collection procedure specific to each protocol, network packet data can ultimately be gathered and saved in either pcap or pcapng file extensions.

3.2 Extract features

From each collected packet, a total of 7 features are extracted: sender's identifier (s), receiver's identifier (d), additional information (a), protocol name (p, P), packet length (L), packet time interval (T), and packet transmission direction (D). Features abbreviated in lowercase are values used for clustering packets that constitute a communication session, while those abbreviated in uppercase are essential features used for device identification from the relationships of clustered packets.

```
Flow Generator():
for each packet in pcapng file:
    source_id ← packet.source_id
    destination_id ← packet.source_id
    additional_info ← packet.additional_info
    protocol ← packet.protocol
    length ← packet.length
    timestamp ← packet.timestamp

    for each flow in all flows:
        if flow.sid == source_id and flow.did == destination_id:
            if flow.ai == additional_info and flow.protocol == protocol:
                timestamp ← flow.start_time − timestamp
                flow.add_value((timestamp, length, direction ← 1))
                break
        else if  flow.did == source_id and flow.sid == destination_id:
            if flow.ai == additional_info and flow.protocol == protocol:
                timestamp ← flow.start_time − timestamp
                flow. add_value((timestamp, length, direction ← 0))
                break
    else:
        add new flow
        flow.sid ← source_id
        flow.did ← destination_id
        flow.ai ← additional_info
        flow.protocol ← protocol
        flow.start_time ← timestamp
        flow. add_value((timestamp ← 0, length, direction ← 1))
```

Figure 3: Pseudocode for flow object generation

Figure 3 illustrates the pseudocode for creating the flow object proposed in this study for device identification. This object was formulated by partially leveraging the contents proposed by Ola Salman et al[7]. A flow is an object comprised of key-value pairs where the key takes the form of a tuple. This tuple consists of the sender device's identifier, the receiver device's identifier, the type of protocol, and a field recording additional information. Such supplementary details refer to communication data beyond device identifiers like IP or Node ID. When identifiers are identical due to a specific field in the packet, additional procedures are required for processing. For tuples extracted from two different packets, they are considered identical under the following conditions:

- Are the protocol details of the two packets the same?
- the sender device identifiers and receiver device identifiers of the two packets either identical or in reverse order?

The tuple to be used as the value for the flow is also extracted from the header information. Each tuple incorporates the packet timestamp's margin of error, the length of the packet, and details about the transmission direction. The transmission direction is determined based on the relationship of the sender and receiver stored in the key tuple and the relationship of the sender and receiver in the packet from which the value is being extracted. For instance, if the sender's identifier in the key tuple matches the sender's identifier in the packet, and the receiver's identifier in the key tuple corresponds to the receiver's identifier in the packet, then this value is set to 1. Conversely, if the sender's identifier in the key tuple matches the receiver's identifier in the packet, and the receiver's identifier in the key tuple matches the sender's identifier in the packet—essentially the reverse of the previous situation—the value is set to 0.

3.3 Refine data to Flow objects

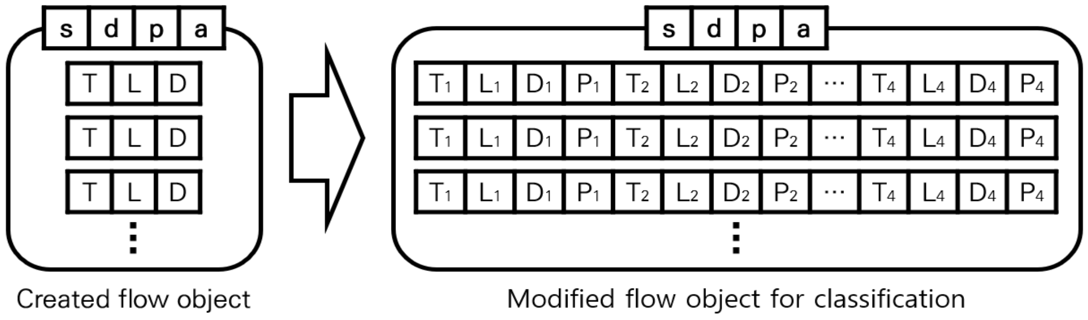
s d p a
T L D
T L D
T L D
Created flow object
s d p a
T1 L1 D1 P1 T2 L2 D2 P2 ··· T4 L4 D4 P4
Modified flow object for classification

Figure 4: Modification of flow object values for classification

To create a flow object, preprocessing is performed as shown in Figure 4. Among them, clustering is performed based on the sender's identifier, receiver's identifier, additional information, and protocol name. In this experiment, since the protocol used was limited to Zigbee, the additional information was set as the PAN ID. Clustered packets additionally possess three features regarding each packet's length, time interval, and transmission direction. The values of the three features extracted from the packets composing the session can be seen as the features of that session. Additionally, even if they are different protocols, there may be sessions with similar three features, so the protocol can also be treated as a feature of that session. Thus, the four features extracted from the packets composing the communication session can be seen as the session's features. However, the time interval feature and transmission direction feature of the first packet composing each session are always created identically, so classification cannot be conducted properly with just four features. To address this issue, features extracted from four consecutive packets are serialized to transform into a total of 16 features for classification.

3.4 Identify Internet of Things Devices

Based on the acquired Flow Objects through the aforementioned processes, various machine learning algorithms are employed to learn and categorize which type of IoT device constitutes the given Flow Object. In most scenarios, this situation is treated as multiclass classification, with algorithms such as Decision Tree and Random Forest being commonly utilized. Additionally, depending on the specific network environment, selecting an appropriate algorithm beyond these can yield higher accuracy.

4 Device Identification and Classification

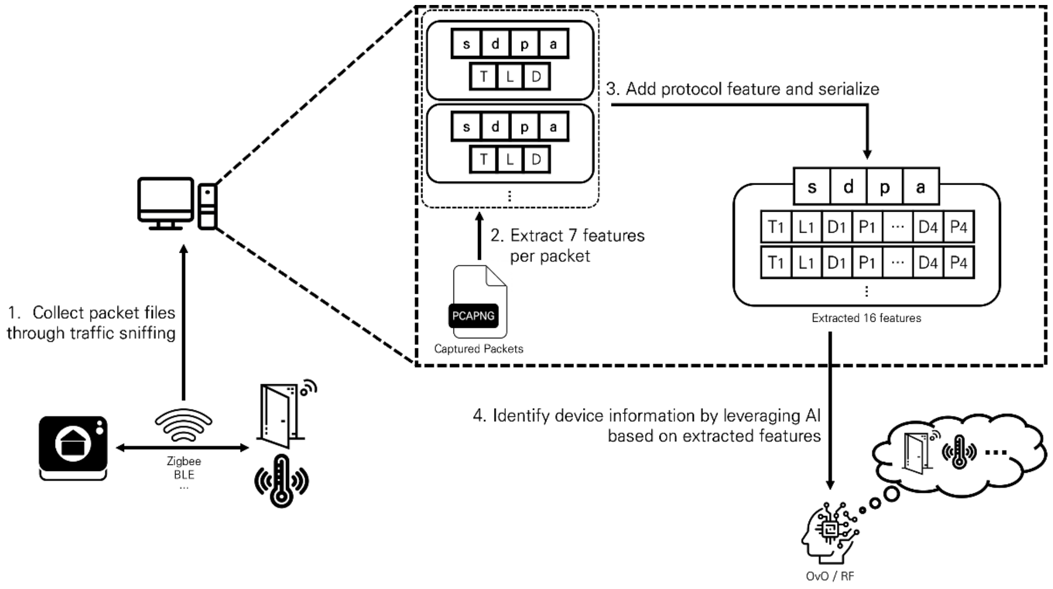
3. Add protocol feature and serialize
2. Extract 7 features per packet
PCAPNG
Captured Packets
Extracted 16 features
1. Collect packet files through traffic sniffing
Zigbee
BLE
...
4. Identify device information by leveraging AI based on extracted features
OvO / RF

Figure 4: Diagram for device identification experiment

In this chapter, we introduce a case study that implements the proposed model presented in Chapter 3 and measures the actual performance of device identification. Figure 4 summarizes the experimental process graphically. Firstly, packets between IoT devices communicating using the Zigbee protocol are sniffed. Then, using the Wireshark tool, they are converted into pcapng files and saved on the analysis PC. Subsequently, seven features are extracted from each collected packet. Based on the extracted features, Flow Objects are generated for use in machine learning algorithms. These created Flow Objects, along with label data, are input into appropriately structured classification algorithms for training. We plan to assess the performance of the trained machine learning model to determine the effectiveness of the approach proposed in this study.

4.1 Identify Internet of Things Devices

Table 2: List of devices to configure the experiment

	Device Model
1	Aqara Door Sensor
2	Aqara Switch
3	Aqara Temperature/Humidity Sensor
4	SmartThings Multipurpose Sensor

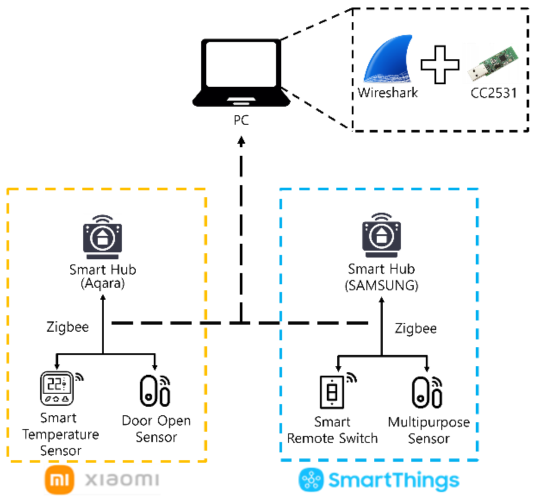

Figure 4: Experimental setup for device identification experiment

Table 3: List of devices and labels to be identified

	Device Model	Type	Vendor
1	AeoTec Button	Sensor	AeoTec
2	AeoTec Motion Sensor	Sensor	AeoTec
3	AeoTec Multipurpose Sensor	Sensor	AeoTec
4	AeoTec Water Leak Sensor	Sensor	AeoTec
5	Aqara Switch	Sensor	Aqara
6	Aqara Door Sensor	Sensor	Aqara
7	Aqara Temperature/Humidity Sensor	Sensor	Aqara
8	Philips Hue White	Actuator	Philips
9	Sengled Smart Plug	Actuator	Sengled
10	SmartThings Button	Sensor	Samsung
11	SmartThings Multipurpose Sensor	Sensor	Samsung
12	SmartThings Smart Bulb	Actuator	Samsung
13	Sonoff Smart Plug	Actuator	Sonoff

Table 2 displays a list of IoT devices that make up the data collection environment, providing details about four devices connected to both the Aqara Hub M1S and Samsung SmartThings Hub. Zigbee communication data between each hub and the associated devices were gathered over 10 days. For the data collection process, a laptop equipped with the Zigbee collection tool, CC2531, was positioned between the hub and each device. Packet data was then saved in the form of pcap files using the packet monitoring software, Wireshark. Furthermore, the publicly available IoT network dataset, CIC IoT 2022[10], was employed to incorporate information from a broader range of devices and network traffic. This dataset was designed for various purposes and consists of packet files for multiple protocols such as Zigbee, Z-Wave, HTTP, TCP, and UDP. In this study, only information concerning devices using the Zigbee protocol was utilized. Data from a total of 16 devices, obtained from the dataset, was integrated with the information gathered in the experimental setup for further experiments. Labels for each device under investigation were set as shown in Table 3. The device model name was labeled to represent the device's product name, while the device type was categorized and labeled mainly into sensors and actuators. The vendor of the device was set to the name of the company that sells or manufactures the device.

Table 4 shows the number of communication packets for each device included in the training set, test set, and validation set. Unlike the four devices for which a direct experimental environment was set up to capture packets, the network data collected from CIC IoT 2022 could not be further collected or modified, resulting in a relatively smaller dataset.

Table 4: The number of packets included in each set

	Device Model	Number of Packets		
		Train	Test	Validation
1	AeoTec Button	212	211	70
2	AeoTec Motion Sensor	211	214	66
3	AeoTec Multipurpose Sensor	1352	472	113
4	AeoTec Water Leak Sensor	399	205	246
5	Aqara Switch	2583	9489	1897
6	Aqara Door Sensor	57	228	41
7	Aqara Temperature/Humidity Sensor	160	564	89
8	Philips Hue White	105	112	34
9	Sengled Smart Plug	151	200	59
10	SmartThings Button	237	213	53
11	SmartThings Multipurpose Sensor	1749	6967	1140
12	SmartThings Smart Bulb	540	395	142
13	Sonoff Smart Plug	102	105	23

4.2 Extract features

Based on the collected data, we extracted 7 features using the method mentioned in Chapter 3.2. Feature extraction was conducted from packets corresponding to the ZBEE_NWK layer, and among them, the feature for the additional information section was set as the PAN ID. Subsequently, based on the features corresponding to s, d, p, and a, packets forming the same communication session were clustered to create a Flow object.

4.3 Refine data to Flow object

For the packets comprising the created Flow object, as described in Chapter 3.3, four packets were serialized to produce 16 features, with T, L, D, and P being repeated four times. The 16 values were subsequently used as features for the machine learning-based classification algorithm.

4.4 Identify Internet of Things Devices

Based on the collected flow objects, three device identification experiments were conducted: identification of device type, device vendor, and device model. Classification algorithms such as DT(Decision Tree) and RF(Random Forest) were utilized for identification. Additionally, deep learning techniques, RNN(Recurrent Neural Network), and LSTM(Long-Short Term Memory) were leveraged to attempt device identification.

The parameters and the range of hyperparameters used for training the models were set as shown in Table 5. Models based on Decision Tree (DT) and Random Forest (RF) were optimized through GridSearchCV, while models based on Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) were configured through Hypermodel.

Table 5: Range of parameters/hyperparameters used

Algorithm	Parameter/ Hyperparameter	Range
RF	n_estimators	200-500
	max_depth	5-30
	min_samples_leaf	2-10
	min_samples_split	2-10
	max_features	auto, sqrt, log2
DT	max_depth	5-30
	min_samples_leaf	2-20
	min_samples_split	2-20
	max_features	auto, sqrt, log2
RNN	units	64-256 (steps of 16)
	num_layers	1-3
	dropout rate	0.1-0.3 (steps of 0.1)
LSTM	units	64-256 (steps of 16)
	num_layers	1-3
	dropout rate	0.1-0.3 (steps of 0.1)

5 Results

Table 6: Results of the device model name identification experiment

Algorithm	Evaluation Metrics			
	Accuracy	Precision	Recall	F1-Score
RF	**93.89%**	0.8478	0.7259	0.7538
DT	93.82%	0.8333	0.7208	0.7432
RNN	83.80%	0.8278	0.8380	0.8180
LSTM	81.41%	0.7914	0.8141	0.7893

Table 7: Results of the device type identification experiment

Algorithm	Evaluation Metrics			
	Accuracy	Precision	Recall	F1-Score
RF	97.90%	0.9815	0.9734	0.9772
DT	**98.05%**	0.9824	0.9756	0.9788
RNN	90.08%	0.9068	0.9008	0.8018
LSTM	87.71%	0.8806	0.8771	0.8780

Table 8: Results of the device vendor identification experiment

Algorithm	Evaluation Metrics			
	Accuracy	Precision	Recall	F1-Score
RF	**94.76%**	0.8947	0.7315	0.7798
DT	94.64%	0.8967	0.7308	0.7774
RNN	89.82%	0.8891	0.8982	0.8898
LSTM	85.69%	0.8324	0.8569	0.8389

The results of each identification experiment are depicted in Tables 6, 7, and 8. For the model that identifies device model names, an accuracy of 93.89% was achieved using the RF algorithm. However, the experiment identifying the model name displayed the lowest accuracy. This is believed to be because, among the three kinds of experiments, this one had the highest label diversity. For the same reason, the experiment identifying the device type, that had the least label diversity, was observed to display a relatively high accuracy. For the experiment identifying the device type, an accuracy of 98.05% was achieved using the DT algorithm. Lastly, for the experiment identifying the device manufacturer, an accuracy of 94.76% was achieved using the RF algorithm.

In conclusion, it was confirmed that utilizing RF and DT allowed for the highest classification performance. Notably, they achieved higher accuracy than RNN and LSTM, which specialized in processing time-series data. However, the two deep learning-based classification techniques can vary significantly depending on their structure and complexity. Therefore, further research enhancing the diversity of model parameters appears to be necessary. Also, some models composed of DT and RF achieved lower F1-Scores compared to those constructed with RNN or LSTM. This indicates that even if models built with RF and DT have high Accuracy, they might have lower Precision and Recall, suggesting that there's room for improvement in models based on these two algorithms.

6 Conclusion

As the scope of IoT services expands with the advancement of technology, the incidence of IoT security intrusions is also increasing. From the perspective of intrusion investigation, the device identification phase is one of the critical tasks for identifying the source of information for digital forensics. However, due to the diversity of devices and encrypted communication, device identifica7tion based on network traffic is a challenging task. In this study, we proposed a method to create a flow object by extracting identifiable information from the unencrypted header area and an IoT device identification solution based on flow objects using a machine learning classification model. As a result of conducting device identification experiments after setting up an actual IoT device environment, it was confirmed that it could achieve an accuracy of up to 98.05%. The dataset and code used in this experiment are available in the GitHub repository [11].

The flow object presented in this study was designed and implemented not to be based on a specific protocol. However, the network data collected in this study consists of Zigbee network packets, necessitating additional experiments for supplementation. Therefore, we plan to conduct further studies targeting additional protocols such as TCP/IP and Z-Wave in the future study.

Acknowledgement

"This work was supported by Institute for Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT)(No.2022-0-01203, Regional strategic Industry convergence security core talent training business)

The results of a study on the supported by Nuclear Safety Research Program through the Korea Foundation of Nuclear Safety (KoFONS) using the financial resource granted by the Nuclear Safety and Security Commission (NSSC) of the Republic of Korea (No.2106061).

References

[1] Kent, K., Chevalier, S., & Grance, T. (2006). Guide to integrating forensic techniques into incident. Tech. Rep. 800-86.
[2] Sarhan, S. A. E., Youness, H. A., & Bahaa-Eldin, A. M. (2023). A framework for digital forensics of encrypted real-time network traffic, instant messaging, and VoIP application case study. Ain Shams Engineering Journal, 14(9), 102069.
[3] Mazhar, N., Salleh, R., Zeeshan, M., & Hameed, M. M. (2021). Role of device identification and manufacturer usage description in iot security: A survey. IEEE Access, 9, 41757-41786.
[4] He, S., & Chan, S. H. G. (2015). Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons. IEEE Communications Surveys & Tutorials, 18(1), 466-490.
[5] International Organization for Standardization(2022). IoT Security and privacy(ISO Standard No. 27400:2022). https://www.iso.org/standard/44373.html
[6] Jafari, H., Omotere, O., Adesina, D., Wu, H. H., & Qian, L. (2018, October). IoT devices fingerprinting using deep learning. In MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM) (pp. 1-9). IEEE.
[7] Salman, O., Elhajj, I. H., Chehab, A., & Kayssi, A. (2022). A machine learning based framework for IoT device identification and abnormal traffic detection. Transactions on Emerging Telecommunications Technologies, 33(3), e3743.
[8] Babun, L., Aksu, H., Ryan, L., Akkaya, K., Bentley, E. S., & Uluagac, A. S. (2020, June). Z-iot: Passive device-class fingerprinting of zigbee and z-wave iot devices. In ICC 2020-2020 IEEE International Conference on Communications (ICC) (pp. 1-7). IEEE.
[9] Zhang, J., Li, X., Li, J., Dai, Q., Ling, Z., & Yang, M. (2023). Bluetooth Low Energy Device Identification Based on Link Layer Broadcast Packet Fingerprinting. Tsinghua Science and Technology, 28(5), 1-11.
[10] GitHub. AI_001. Available online: https://github.com/pezluna/AI_001