Hindawi
Security and Communication Networks
Volume 2021, Article ID 9069123, 16 pages
https://doi.org/10.1155/2021/9069123

Research Article

WND-Identifier: Automated and Efficient Identification of Wireless Network Devices

Fangzhou Zhu, Liang Liu, Simin Hu, Ting Lv, and Renjun Ye

College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China

Correspondence should be addressed to Liang Liu; liangliu@nuaa.edu.cn

Received 3 August 2021; Revised 22 October 2021; Accepted 30 October 2021; Published 20 November 2021

Academic Editor: Marimuthu Karuppiah

The widespread application of wireless communication technology brings great convenience to people, but security and privacy problems also arise. To assess and guarantee the security of wireless networks and user devices, discovering and identifying wireless devices become a foremost task. Currently, effective device identification is still a challenging issue, as device fingerprinting requires huge training datasets and is difficult to expand, and rule-based identification is not accurate and reliable enough. In this paper, we propose WND-Identifier, a universal and extensible framework for the identification of wireless devices, which can generate high-precision device labels (vendor, type, and product model) efficiently without user interaction. We first introduce the concept of device-info-related network protocols. WND-Identifier makes full use of the natural language features in such protocol messages and combines with the device description in the welcome page, thereby utilizing extraction rules to generate concrete device labels. Considering that the device information in the protocol messages may be incomplete or forged, we further take advantage of the application logic independence and stability of the device-info-related protocol, so as to build a multiprotocol text classification model, which maps the device to a known label. We conduct experiments in homes and public networks and present three application scenarios to verify the effectiveness of WND-Identifier.

1. Introduction

Nowadays, there are portable devices accessing the Internet through WLAN in every corner of the world. Compared with another common wireless network technology—cellular network, Wi-Fi has advantages in terms of ease of deployment and low cost. Therefore, in the deployment of indoor and private local networks, Wi-Fi is still the preferred technology that provides reliable and convenient support for home network devices [1], from portable devices such as smartphones and laptops to smart home devices such as webcams, speakers, and thermostats. The sixth-generation Wi-Fi technology brings higher data rates, more wireless capacity, and lower power consumption [2]. According to related predictions [3], the global Wi-Fi market will grow from USD 9.4 billion in 2020 to USD 25.2 billion in 2026.

With the rapid increase of Wi-Fi devices, security and privacy issues also arise. In 2017, the key reinstallation attack (KRACK) [4] took advantage of design flaws in the WPA/WPA2 encryption protocol, allowing attackers to hijack TCP connections and steal user privacy. In the year 2019, the Kr00k vulnerability [5] affected devices using Cypress and Broadcom Wi-Fi chips, making more than one billion Wi-Fi devices such as access points, smartphones, tablets, and IoT gadgets vulnerable to attacks.

For network administrators of corporate networks or public Wi-Fis, there may be security risks in the managed network. Device vendors may lag behind in pushing security updates, while users normally lack security awareness or necessary skills and do not apply security patches in time, which makes the devices vulnerable to attacks. In addition, there can also be unauthorized malicious devices. If employees privately connect IoT devices with security risks to the enterprise network without authorization, cross-infection may occur [6, 7]. Adversaries can take control of the devices in the WLAN, thereby stealing users' sensitive information and endangering the security of the network.

Therefore, there is a need for an effective solution to discover and identify vulnerable and unauthorized devices and evaluate the security of the entire network.

For ordinary Wi-Fi users, there is a need to determine the security of the network they connect to and ensure personal privacy and data security. As shown in Figure 1, we imagine a scenario where a user connects a portable device to a wireless network, which may be public Wi-Fi located in an airport, hotel, restaurant, etc. In general, determining whether the wireless network is secure is a hard task for the user. For example, the wireless router the user connected to may exhibit known exploitable security vulnerabilities due to the lag in the firmware update. In particular, the emergence of Evil-Twin attacks further aggravates privacy threats. The adversary creates an unauthorized rogue access point (AP), which has the same service set identifier (SSID) as the legitimate router and provides higher signal strength, to hijack the wireless connection from the client [8], and then eavesdrop on the network traffic. Furthermore, there may be hidden malicious devices in the network, such as wireless cameras and listening devices, which can be used by adversaries to monitor user behavior and steal sensitive data. Hence, users require a feasible solution to evaluate the security of wireless APs when accessing the network, discover, and identify hidden malicious devices in the WLAN and protect their own security and privacy from infringement.

Therefore, in order to evaluate and protect the security of the management network and assess the risk of the public network users connect to, it is essential to discover and identify wireless devices, so as to further track, monitor, and apply protective measures. In the literature, device fingerprinting is the most common method of identifying specific types of devices. The software and hardware features of the device are usually utilized to realize the identification of the operating system [9–12], browser [13–16], and wireless access point [17–20]. However, the above schemes are tailored for specific device types, which are not universal and extensible.

To realize the general identification of wireless devices, network traffic characteristics are extracted to establish a classification model. This kind of methods includes active identifications that elaborately construct unique detection requests [21, 22] and passive schemes for monitoring and analyzing network traffic [23–30]. Nevertheless, the demand for a large amount of training data sets is difficult to adapt to the rapidly growing network devices. In addition, the above methods cannot provide concrete device information, resulting in limited application scenarios. By contrast, rule-based identification schemes [21, 31–37] extract specific identification information (e.g., device vendor and product model) from the response data in the application layer to label network devices. However, the above solutions need to send requests actively, which can be easily blocked by firewall rules. Moreover, the device text information is single-source, which is only grabbed from the response data and may be incomplete and vulnerable to forgery and deception.

In this paper, we try to solve the above problems. We propose a universal and extensible framework for automated identification of Wi-Fi devices, called WND-Identifier.

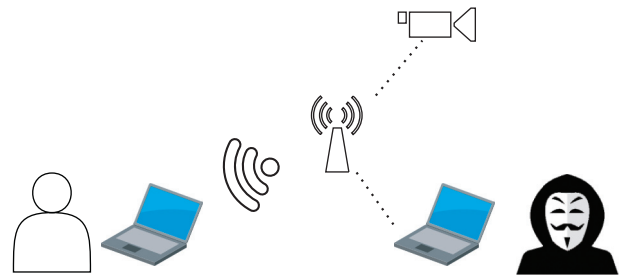


Figure 1: Insecure wireless network.

According to the needs of network administrators and users, we define the identified device label as (vendor, device type, and product model). In this way, network administrators can confirm the identities of connected devices, configure firewall rules, and scan for known vulnerabilities with the model entities. Users can identify rogue APs and hidden devices in the network by their types. The main challenge of this work is the wide variety and continuous growth of wireless devices. They are produced by different manufacturers, have distinct models or firmware, and apply diverse combinations of network protocols. Therefore, it is an arduous task to find a general and high-precision solution to overcome the weaknesses of traditional identification methods.

The core idea of WND-Identifier is to utilize natural language features in network data packets. To provide basic services, Wi-Fi devices require to apply a set of necessary broadcast and multicast protocols. For example, the devices obtain IP addresses by broadcasting DHCP discovery and request messages and utilize the mDNS protocol to send multicast query messages for hostname resolution. We observe that, in such broadcast and multicast data packets of Wi-Fi devices, some header and data fields often contain implicit and explicit device-specific feature information. Therefore, we aim to extract and parse effective information from various protocol packets to identify devices in detail, including mDNS, DHCP, DHCPv6, SSDP, BROWSER, NBNS, and LLMNR. We define these protocol messages and their key fields as device-info-related. It is worth noting that WND-Identifier can run on any user's computer in the WLAN without sending any probes actively during the entire identification process. The above device-info-related protocols are mainly broadcast and multicast protocols, which can be received by any user in the network, and there is no need to set the network card to promiscuous or monitoring mode. Hence, the entire process of device identification is passive and nonintrusive.

WND-Identifier includes three stages: data collection, rule-based device identification, and text classification-based device identification. At first, we capture network traffic in the WLAN and screen out device-info-related protocol packets. Next, the rule-based device identification component analyzes and extracts device labels of Wi-Fi devices, including the vendor, type, and product model. Some protocol fields in the data packets contain implicit and explicit device-related description texts (e.g., the Qname of mDNS, User-Agent of SSDP), from which we can directly

extract or indirectly infer device labels. For some devices with startup screens or management websites like wireless routers and IP cameras, manufacturers often hard-code device information on the homepage. We establish an HTTP connection with the webserver, grab, and analyze the source code of the device's web page. Utilizing prebuilt extraction rules, we identify the device entity in the web page automatically.

In the above process, we preliminarily generate the device label in the form of (vendor, type, model), but the device information may sometimes be unreliable. For example, the field in the data packet is easy to forge, and the device information may be incomplete and only includes coarse-grained information at the vendor level. In this case, we require to further identify the device and confirm the legitimacy and reliability of the information. Different from unicast communication related to specific applications (such as file transfer and instant messaging), the broadcast and multicast protocols we leverage are usually used for service discovery and device configuration and are not related to specific application logic and user interaction behavior, so the traffic is relatively stable and contains highly device-specific information. We need to aggregate network data packets from the same device according to the protocol type and observe that the control and configuration protocol packets from the same device tend to have similar text content in different time intervals, while distinct types of devices are often quite different. As a result, we establish a similarity model to compare network traffic from different Wi-Fi devices. It is worth noting that some of the information is not recognizable to human analysts, but we treat it directly as natural language text, which can also be utilized to distinguish different devices. For various network protocols, we apply K-nearest neighbors (KNN) classifier to achieve further effective classification and identification of Wi-Fi devices.

In this paper, we present WND-Identifier, a novel, universal, and extensible scheme of device identification. Our key contributions can be summarized as follows:

(i) *Rule-Based Device Identification.* We introduce the concept of device-info-related protocols. Using the natural language features of the key fields in this type of protocol packets, combined with the device description information in the welcome interface, we establish extraction rules to parse implicit and explicit device information, including vendor, type, and product model, thus realizing automated identification of Wi-Fi devices.

(ii) *Text Classification-Based Device Identification.* By analyzing configuration and control data packets that are irrelevant to specific application logic, we utilize TF-IDF to generate word vectors and perform similarity calculations for data fields from various devices and use the KNN algorithm to build classifiers for different protocol types, thereby achieving accurate and efficient device identification.

(iii) *Effectiveness.* We utilize closed and open-world data sets to simulate the home network and public Wi-Fi environment, respectively, and conduct experiments to verify the effectiveness of WND-Identifier, which achieves precise identification of wireless network devices with only a small amount of traffic for a few minutes.

The rest of this paper is organized as follows. Section 2 surveys the background of device identification, and then we give the threat model and preset application scenarios in Section 3. In Section 4, we describe the design and implementation of device identification technology of WND-Identifier in detail. We conduct experiments in Section 5 and discuss some issues. Finally, we conclude in Section 6.

2. Device Identification

Device identification aims to make full use of the characteristics of the target device or acquire and extract effective information with high identification [38], regardless of the information that is easy to forge and modify (e.g., IP address and MAC address) [27], to establish a unique device identifier and break the anonymity of the device, thereby realizing the discovery, labeling, and monitoring of the target device. The current solutions used to identify user equipment in the network mainly include device fingerprinting based on software and hardware characteristics, device identification based on network traffic features, and rule-based identification.

2.1. Device Fingerprinting. Device fingerprinting is a technology that takes advantage of implementation differences in standard specifications to generate device-specific signatures and identify individual devices [39, 40]. Early fingerprinting usually uses hardware and software characteristics to identify a specific type of device. CryptoFP [41] utilizes the difference of the internal clock signal to distinguish computers by calculating the execution time of the instruction sequence. PRAPD [20] clusters the RSS vectors from different wireless APs by measuring the signal strength of the beacon frame, and realizes the detection of malicious APs. Radio frequency (RF) fingerprinting [42, 43] uniquely associates wireless devices with a given transmitter through the hardware defects of RF transmitters, breaking the anonymity of users in wireless communication systems. Browser Fingerprinting [13–16] implements a stateless tracking technology that uniquely identifies user equipment by using features such as user agents, HTTP headers, fonts, and extensions installed on the computer. Traditional device fingerprinting usually identifies a single specific type of devices, resulting in limited versatility and scalability.

2.2. Device Identification Based on Traffic Features. In order to achieve effective classification and management of network devices, traffic patterns become available distinguishing features. Device identification based on network traffic characteristics can be divided into two categories:

active identification of establishing a connection with the targeted device and sending requests to collect response packets and passive scheme that monitors the network traffic silently [44]. Nmap [21] is the most widely used active identification tool, which constructs unique detection requests and extracts the characteristics of the response data to identify the vendor and operating system of the remote host. Bratus et al. [22] construct a frame generator and injector to identify rogue devices by observing the response of the wireless device to a series of nonstandard and malformed 802.11 frames.

Active schemes have defects such as excessive overhead, low efficiency, and easy prevention by intrusion detection systems (IDSs). Therefore, nonintrusive methods of passively observing and analyzing network traffic are often more effective. Gao et al. [23] capture the egress traffic of wireless APs and apply wavelet analysis to the packet interarrival time to identify unknown APs. DeWiCam [24] explores the video and audio traffic patterns of IP cameras and realizes their classification and the detection of hidden cameras. WDMTI [25] extracts option codes and request parameters from DHCP traffic and uses Hierarchical Dirichlet Process (HDP) to identify manufacturers and types of mobile and IoT devices. AUDI [26] models the periodic communication traffic of IoT devices to perform identification and generate abstract device type labels. GTID [27] applies statistical techniques to network traffic, creates unique device and type signatures, and uses artificial neural networks (ANN) for classification. Sivanathan et al. [28] collect statistics on traffic from the aspects of activity patterns, signals, protocols, etc., distinguish IoT devices in smart cities and campus networks, and identify them uniquely. OWL [29] acquires broadcast/multicast traffic, extracts fields in multiple sets of protocols, applies multiview wide and deep learning solutions to perform device identification and malicious device detection. IoT Sentinel [30] extracts features from the burst traffic in the device setup phase and uses random forest classifiers to identify the type of IoT devices. However, due to the various types and models of portable and IoT devices, a huge data set needs to be collected and trained to achieve accurate classification. It is an arduous task to adapt to the demand for the rapid growth of the devices. Furthermore, the classification label of the device is abstract (e.g., Type #1, Product #2), so that the specific correspondence needs to be manually identified by the administrator in advance, which hinders its automation and further expansion.

2.3. Rule-Based Device Identification. Rule-based device identification is a process of directly analyzing and extracting concrete information (e.g., device type, vendor, and product model) from the response data of the application layer to label devices according to predefined rules. Banner grabbing is the most widely used solution, which utilizes FTP, HTTP, Telnet, SSH, and other protocols and proprietary programs to establish a connection with the remote host [45], observes, and analyzes the response data to identify network devices. Nmap [21], Masscan [31], and ZMap [32] are all tools for quick scanning of Internet IP ports, collecting banner information from different device ports, and analyzing running systems and services. Shodan [33] and Censys [34] are search engines used to monitor and manage IP devices on the Internet. After performing SYN scanning to find the list of open ports on the target device, they apply banner grabbing on these ports, and the scan results include information about the device, service, and its version running on the port [46].

In traditional schemes, the establishment of extraction rules often needs to be done manually and requires rich experience accumulation, which leads to excessive manpower and time costs. To overcome the above problems, ARE [35] establishes automated acquisitional rules to capture specific identifications including type, vendor, and model from the response data of IoT devices. Yang et al. [36] collect information from the Internet to build a complete product database of IoT devices and combine rules to label network devices. WNV-Detector [37] designs general rules for accessing device webpages and parse out fine-grained information. However, the response data of network devices often contain only partial information, limiting the effectiveness and coverage of the identification. Also, textual information is easy to forge and makes it hard to distinguish possible deceptive behaviors. Moreover, the active identification can be easily blocked by IDSs and firewall rules.

3. Threat Model and Application Scenarios

In this section, we discuss the threat model and preset the application scenarios of WND-Identifier.

3.1. Threat Model. We assume that the adversaries conduct attacks in the WLAN, and the specific behaviors can be (1) connecting unauthorized devices to the wireless network such as hidden cameras and bugs [24, 47]; (2) using IoT honeypots or virtual machines to forge legitimate real devices [48, 49]; (3) utilizing software to counterfeit the wireless AP, which has the same SSID and MAC address as the legitimate one, deceiving the user to connect and further performing malicious attacks [17–20]; (4) exploiting the device vulnerabilities that have been disclosed on the Internet to endanger the security of user devices and the entire platform [37, 50].

3.2. Application Scenarios. WND-Identifier is designed to identify devices in detail in a WLAN. To illustrate our motivation, we preset the following three application scenarios. Figure 2 shows an application example of device identification, including the unified maintenance and management of devices and the detection of abnormal devices.

3.2.1. Network Administrator. In public Wi-Fi, enterprise, or home networks, network administrators need to discover and identify all the connected devices in the WLAN,

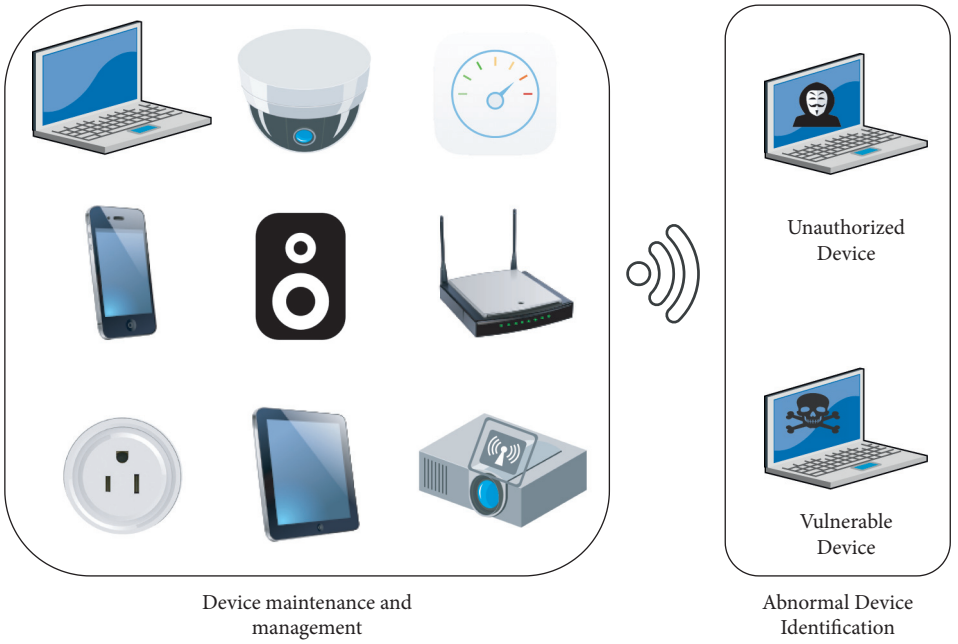


Figure 2: Application scenario.

so as to distinguish legal and unauthorized malicious devices. Traditional network access control schemes maintain whitelists based on MAC addresses and IP addresses, but these identifiers are extremely vulnerable to modification and spoofing by attackers [51,52]. In contrast, WND-Identifier aims to utilize diversified device information, which greatly improves security and realizes reliable management and maintenance of network devices. Furthermore, targeted security policies can be applied to protect the security of connected devices and the entire network. For example, we can configure firewall rules for different device types contrapuntally and scan the known security vulnerabilities of the devices by searching the Internet vulnerability database with the concrete device model information [37].

3.2.2. Wi-Fi User. For ordinary users connected to the network, it is vital to confirm that the connected wireless AP is not forged by an adversary. Furthermore, users can utilize device identification to discover malicious devices hidden in the network, such as IP cameras, bugs, and virtual machines that are faked as real devices so as to protect their own security and privacy from infringement.

3.2.3. Network Security Software and Platform. Network security software (e.g., AVG, Avast, Kaspersky, and Norton) installed on the administrator's PC or user's smartphone needs to identify and manage devices with various types and vendors from different users, scan for security vulnerabilities, check firmware updates, monitor and protect sensitive data, and discover the abnormal behavior of illegal device access, thereby ensuring the security of user devices and entire networks.

4. WND-Identifier

To achieve accurate and efficient identification of Wi-Fi devices, we propose a universal, automated, and extensible framework called WND-Identifier. In this section, we introduce the overall architecture and key components of WND-Identifier, including data collection, rule-based device identification, and text classification-based device identification.

4.1. Architecture. WND-Identifier is designed to run on devices connected to the network (such as the administrator's PC or the user's smartphone), and there is no need to set the network card to promiscuous or monitoring mode. Network administrators can use WND-Identifier to identify malicious devices, detect vulnerable devices, and evaluate the security of the entire managed network. When the ordinary user connects to a new hotspot, WND-Identifier ensures that the wireless AP is not forged and scans malicious devices hidden in the network, so as to guarantee the user security and privacy. WND-Identifier is mainly divided into three components. The data collection module drives the network card to capture the traffic in the WLAN, and after preprocessing the data packet, WND-Identifier extracts key information from the device-info-related protocol packets and identifies the equipment concretely, including manufacturer and device model. Considering that the device information in the protocol messages may be incomplete or forged, we further utilize the stability and device-uniqueness of the control and configuration traffic that is irrelevant to specific application logic, so as to model and classify the packet texts from various devices. Thus, the effective identification of Wi-Fi devices can be achieved. The overall architecture of WND-Identifier is shown in Figure 3.

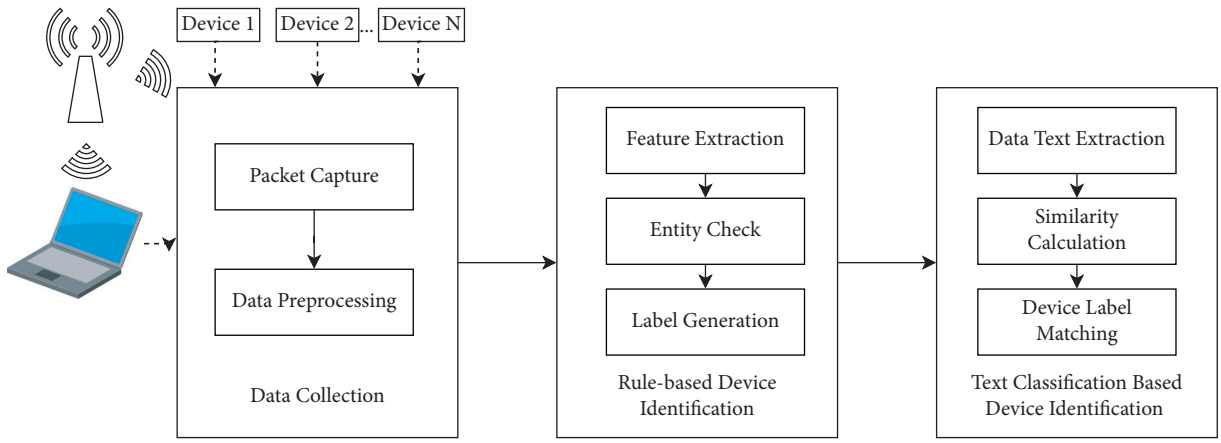


Figure 3: Architecture of WND-Identifier.

4.1.1. Data Collection. The user device connects to the wireless access point and collects data packets in the WLAN, including the private network with administrator rights and the public Wi-Fi as a normal user. It is worth noting that, for open Wi-Fi provided by airports, hotels, and other public places, we can grab network packets without logging in. In practice, we use Tcpdump [53], Wireshark [54], and other packet analysis tools for data acquisition. In the next step, we perform preliminary filtering on the data packets. Our data preprocessing component ignores the traffic that is irrelevant to device identification (such as unicast traffic in the network) and filters the traffic that does not have sufficient natural language characteristics (e.g., ARP packets), thereby screening out device-info-related protocol packets (including DHCP, DHCPv6, mDNS, BROWSER, LLMNR, SSDP, IGMP, NBNS, and other UDP packets). We integrate the packets from devices with the same MAC address and classify them by protocol.

4.1.2. Rule-Based Device Identification. Traditional device fingerprinting schemes are tailored for single specific device types such as browsers [13–16] and wireless APs [17–20], which are not universal and scalable. To achieve general and fine-grained identification of devices in the WLAN, we extract device information in the preprocessed data packets. We classify the preprocessed data packets according to network protocols such as DHCP, SSDP, LLMNR, and pick up device-info-related protocol fields of wireless network devices as features. In addition, we observe that certain IoT devices such as cameras and routers usually provide a welcome page or management website, which has device information hard-coded by the supplier in advance. We access the welcome page according to the intranet IP address, grab the HTML file of the homepage, and apply extraction rules to obtain device information. Finally, the rule-based device identification module generates the device label in the form of vendor, device type, and product model. By comparing with the devices in the whitelist, we can initially identify the unknown and rogue devices in the network.

4.1.3. Text Classification-Based Device Identification. The concrete device label allows us to recognize the identity of the wireless device quickly, but it may be incomplete or forged sometimes. Thus, it is an indispensable task to further utilize features that are highly device-specific and difficult to spoof to identify devices. Our text classification-based device identification component extracts text content from configuration and control data packets that are irrelevant to specific application logic. And, after generating word vectors with TF-IDF, it calculates the similarity between newly connected and whitelisted devices. Then, according to different network protocols, the KNN classifier is used to assign known or new labels to the connected devices. During this period, malicious deception can be discovered through the inconsistency of the output results of the two device identification methods.

4.2. Rule-Based Device Identification. Rule-based device identification aims to make full use of the natural language features in the device-info-related protocol packets to specifically label the device. Previous works use banner grabbing [21, 31–34] and rules [35–37] to extract textual information from devices. However, these solutions need to actively send detections to the device and collect response data for analysis, which is easily blocked by firewall rules. In this paper, we acquire and analyze network data packets passively and design automated extraction rules to parse device information, so as to generate the device labels of Wi-Fi devices.

WND-Identifier utilizes the rule-based device identification to obtain the concrete text information of the wireless network device. We define the device label generated by the process as vendor, device type, and product model. Table 1 gives an example of a set of device labels. The main challenge of this work is that there are many types of portable and home IoT devices, product vendors, and software and hardware designs which are also quite different. We aim to find a universal and automated solution that uniquely labels the user device. The features we use include specific fields in the device-info-related protocol packets and device

information in the welcome interface of Wi-Fi devices. Therefore, the rule-based device identification process can be defined as

$$D_i: \{P_i, W_i\} \Rightarrow (v_i, t_i, m_i). \quad (1)$$

Given a set of devices D_1, D_2, ..., D_m, we utilize the characteristics of the network data packet P_i of the device Di and the web page W_i of the welcome interface, so as to generate device labels including the vendor v_i, device type t_i, product model m_i, and other information.

4.2.1. Feature Selection. The features selected in the rule-based device identification should be universal and independent of the specific device type. In the analysis of actual wireless devices, we find that some fields in the network data packets contain valuable natural language information sufficient to label the equipment. Figure 4(a) shows an example, where the UDP broadcast packet of the device contains specific information about the vendor and type. Moreover, some IoT devices with a welcome interface often display detailed information on the homepage, as shown in Figure 4(b). By checking the homepage source code, the concrete device model can be extracted.

(1) Device-Info-Related Field. In a WLAN, MAC and IP addresses are often used to distinguish devices. However, IP addresses are easy to modify, while researches show that MAC addresses are vulnerable to spoofing and attacks [55, 56]. Therefore, this type of identifier is not included in the characteristics of our rule-based device identification, and only the first 6 hexadecimal digits of the MAC address are extracted to label the manufacturer of the network card (not necessarily the same as the equipment vendor). Through actual observation and analysis, we find that different devices tend to apply distinguishing combinations of protocols. For example, we capture ARP, IGMP, DHCP, mDNS, SSDP, and other protocol packets from the network traffic of a certain camera, which usually contain valuable device information. For the consideration of passive and nonintrusive identification, we do not set the network card to monitor or promiscuous mode or utilize the characteristics of unicast traffic. Table 2 shows some of the device-info-related fields in the network data packets we use in rule-based device identification.

(2) Web Page Description Information. For some IoT devices, e.g., webcams and wireless routers, a lightweight web server with a startup page and some configuration management options is often provided. Network administrators can log in to perform operations such as device maintenance and firmware upgrades. There is often equipment information hard-coded by suppliers on the homepage, such as device type and device models. Any user connected to Wi-Fi can establish an HTTP connection through the device IP address to access the welcome interface without having to obtain administrative rights to log in to the system.

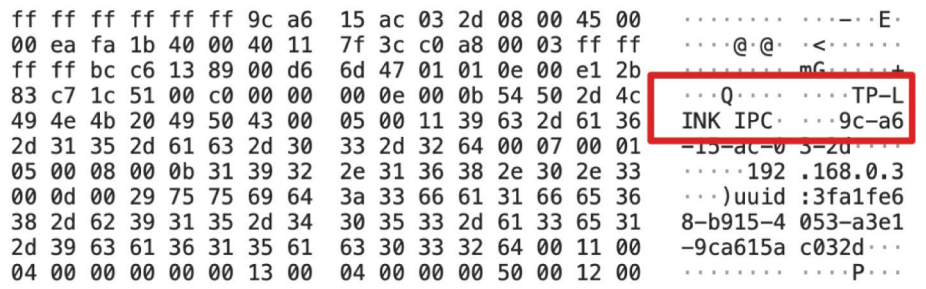
Table 1: Examples of device labels.

Vendor	Type	Product model
Cisco	Router	RV340
Samsung	TV	TU8000
Apple	Smartphone	iPhone12
Honeywell	Thermostat	RTH9585WF1004
HP	Printer	OfficeJet Pro 8025

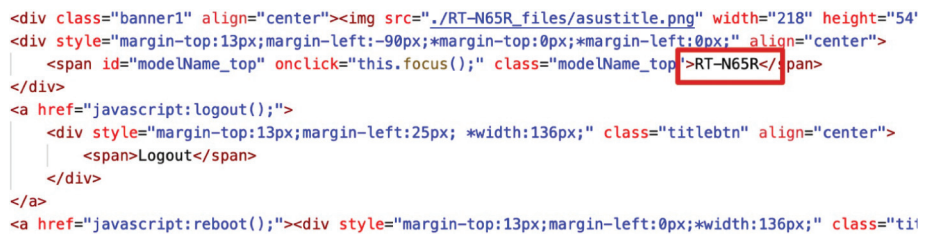
4.2.2. Identification Process. To achieve automated device identification, we capture and analyze the traffic in the wireless network, utilize rules to extract the features of device-info-related fields and web page text information and further verify the validity of the device entity through entity check. The final output is the device label in the form of vendor, type, and product model. Figure 5 shows the entire process of rule-based device identification.

(1) Extraction Rules. Our identification of equipment is based on rules, with vendor, type, and device model as the output. For the device information in the network data packets, we utilize regular rules to filter and extract the field by protocol types listed in Table 2. For the information on the homepage of the device management website, we establish a connection with the webserver based on the device intranet IP address and grab the web page HTML file for analysis. For the identification of device vendors and types, we search the prebuilt database for text matching. Based on sources on the Internet, such as Wikipedia, National Vulnerability Database (NVD) [57], and vendor websites, we collect 670 vendor entities and 33 type entities. For product models, we summarize common matching modes through the analysis of common devices. For example, the title of the web page may directly contain device information, and the model entity can also be extracted from the label text with keywords such as product and model. In most cases, we can achieve accurate information about the device directly. But sometimes, the information contained on the web page may be incomplete, such as lack of a concrete type label. At this point, we can combine the existing collected data and the content on the Internet (such as the manufacturer's website) to indirectly infer and complete the device label. Algorithm 1 presents the process of extracting information from the welcome interface of the device.

(2) Entity Check. After extracting possible entities from the device-info-related protocol fields and device welcome interface, we further verify the validity of the device entities. We observe that the device model is generally composed of alphanumeric characters and may contain special characters such as "-" and "_.". Therefore, we use regular rules like "[A-Za-z] + [-_]?[A-Za-z] ∗ [0-9]+[-_]?[A-Za-z0-9]∗" to filter model labels preliminary. It is worth noting that due to the rapid growth of equipment types, this rule alone cannot ensure coverage of all modes. In the next step, we use vendor and device model as the input for the search engine to retrieve information about the device on the Internet. For example, for a certain wireless router product, we search with the form "search?q = tp-link + wr204&lr = lang_en" to



(a)



(b)

Figure 4: Example of feature selection. (a) Device-info-related-field. (b) Web page description information.

Table 2: Some of the network protocol fields used by rule-based device identification.

Protocol	Feature
DHCP/DHCPv6	Host name
	Vendor class identifier
	Fully qualified domain name
mDNS	Qname
	Rrname
	Rdata
BROWSER	Source name
LLMNR	Qname
SSDP	User-Agent
Other UDP packets	Data
—	First six digits of MAC address

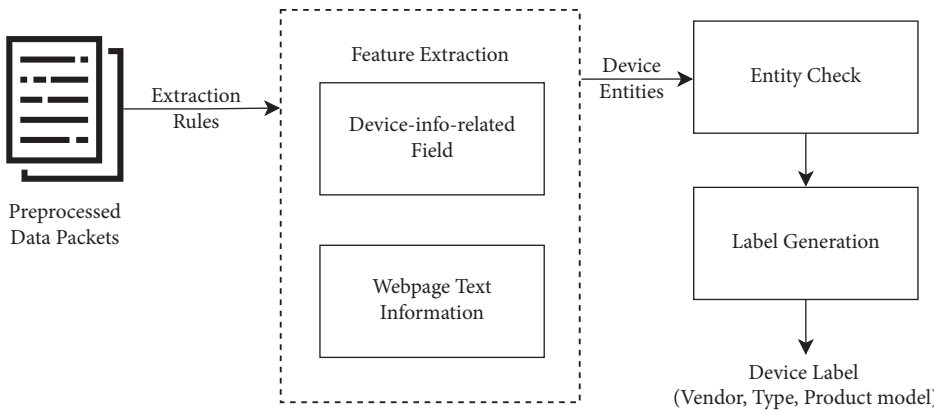


Figure 5: Process of rule-based device identification.

obtain the title and description of the search result and then compare the cosine similarity between the input and the result title. If it is lower than a certain threshold such as 0.4 or the description text does not contain the input device tuple, it is considered that the obtained entity is not the model label, and the inspection of the next item in the list is continued. It is worth noting that the choice of the threshold is obtained in experiments. We use experimental devices (given in Section 5.1) for testing and adjust the threshold to guarantee reliability. To ensure its versatility, we further

Input: ip_address
Output: vendor, type, model
(1) **function** EXTRACT_INFORMATION(ip_address)
(2) page = get_response(ip_address)
(3) vendor = match_vendor_list(page)
(4) type = match_type_list(page)
(5) model_pattern = [title, [class$\bar{p}$ro], [class$\bar{m}$odel]]
(6) possible_models = find_labels(model_pattern)
(7) model = entity_check(vendor, possible_models)
(8) **if** not type **then**
(9) type = indirect_inference(vendor, model)
(10) **end if**
(11) **end function**

Algorithm 1: Extraction of description information from device webpages.

collect device labels on the Internet and input them to the entity check module for testing. When the value is 0.4, all device labels can be covered. In the experiment, we find that if an entity is not a device model, the combination with a legal vendor entity (e.g., Cisco CTU12A) usually results a very low value (such as 0.03). Finally, after screening and testing the acquired entities, we select the most probable product model entity as the output of the rule-based device identification process.

4.3. Text Classification-Based Device Identification. After the rule-based device identification stage, labels in the form (vendor, device type, and product model) are assigned to wireless devices connected to the network. However, concrete device information is sometimes not reliable enough. For example, adversaries can modify some device-info-related fields, and devices in standby mode may not be able to generate enough traffic for concrete identification in a short period of time. Therefore, we further use text classification-based device identification to verify the reliability of the device label and map the connected device to an authorized or unknown one. The whole process includes data text extraction, similarity calculation, and device label matching.

4.3.1. Data Text Extraction. Unlike the personalized and unpredictable unicast traffic, some multicast and broadcast traffic patterns of network devices are usually relatively stable. These network protocols are mainly used for device configuration and control. They are not related to specific application logic and contain highly device-specific features that can be extracted for device identification. We acquire valid text and data fields for different protocols on the preprocessed network packets. For protocols such as mDNS, DHCP, and DHCPv6, we obtain device-info-related fields as Table 2. For protocols such as SSDP, NBNS, BROWSER, LLMNR, and other UDP packets that are not classified as specific application layer protocols, we analyze all the UDP payload content. All extracted field text and data are treated as natural language text. Thus, for a group of devices (d_1, d_2, d_n), after extracting information from the corresponding data packets of the protocol (p_1, p_2, p_m), we can obtain the following data set:

$$X = \begin{pmatrix} d_1p_1 & d_1p_2 & \cdots & d_1p_m \\ d_2p_1 & d_2p_2 & \cdots & d_2p_m \\ \vdots & \vdots & \ddots & \vdots \\ d_np_1 & d_np_2 & \cdots & d_np_m \end{pmatrix}. \quad (2)$$

4.3.2. Similarity Calculation. After the data set is constructed, we need to utilize the natural language features in these texts that are highly relevant to the device to identify the connected Wi-Fi devices. Since the configuration and control traffic pattern of the device is relatively fixed and stable, the similarity comparison of the data text becomes a feasible solution. Our method is based on such observation and assumption: the network packets belonging to the same device that are irrelevant to specific application logic have similar data text according to the protocol type, while the differences between devices from distinguishing vendors and types are relatively large. Different from the text closely related to contextual information, in our application, the value of forms, positions, and contextual semantic of words is limited [58], so the simple and efficient TF-IDF model becomes a suitable scheme. In practice, we first initialize the model and train the data fields of known devices in the whitelist as a corpus. When a new device is connected, we similarly extract the data field, convert it into a sparse vector, and calculate the tf-idf value. By calculating the cosine similarity, we can get the degree of similarity between the same protocol text of the new device and the whitelisted device.

4.3.3. Device Label Matching. Each Wi-Fi device applies a different combination of network protocols. In order to classify data packets of each protocol, we build a KNN model and use the cosine similarity between the data texts calculated by the similarity model as the distance metric. An example of device classification is shown in Figure 6. We construct KNN classifiers according to various protocols and

stipulate that the K value is the number of valid protocols in the data packets obtained from the device. After each protocol classifier votes, the device is assigned the closest label in the whitelist. In the experiment, we find that some unknown devices are classified as legitimate devices with very low similarity. Therefore, we set a threshold for the classifier, and the classification is valid only when the similarity is greater than 0.6. The selection of the threshold is based on a large amount of experimental data. We use experimental devices for testing and divide them into two data sets. We adjust the threshold on one data set and perform repeated tests and use another data set for verification. Thus, it can be confirmed that the value of the threshold is universal. As a result, we achieve further identification and classification of Wi-Fi devices.

4.4. Application. In this section, we present several application scenarios based on WND-Identifier according to the threat model in Section 3.1.

4.4.1. Unauthorized Device Identification. WND-Identifier can establish a whitelist of legal devices in the process of identifying known devices. When a new device is connected to the WLAN, utilizing the natural language features in the configuration and control data packets of the device that are not related to specific application logic, it can be classified as a known legitimate device or an unknown one. In this way, unauthorized devices such as hidden IP cameras can be detected. Moreover, if the adversary attempts to forge an authorized device by modifying the device information, WND-Identifier can utilize the inconsistent information in the identification process to distinguish the illegal device.

4.4.2. Rogue AP detection. The adversary can use computer software and network cards to forge a wireless AP. Since it usually has the same SSID, MAC address, and stronger signal as a legitimate hotspot, the user terminal is likely to connect to the fake AP automatically without permission [59]. When the user connects to the WLAN, WND-Identifier can scan the security of the wireless network and confirm whether the wireless AP is forged by an attacker. It is difficult for a forged soft AP to have the same protocol stack as the real device. For example, the adversary sets up a rogue AP with the same SSID and MAC address as the real device. The user connects to the illegal AP by default as it has a stronger signal and cannot perceive that his/her privacy and data are being violated. However, WND-Identifier can detect the forged identity of the AP from the device-info-related fields of the mDNS data packets. Moreover, in the welcome interface of the real device, WND-Identifier can also acquire the device information hard-coded by the manufacturer that is not available in the phishing web page provided by the rogue AP.

4.4.3. Vulnerable Device Scanning. Existing device identification schemes cannot obtain concrete device information, while our WND-Identifier can generate the device label (vendor, type, product model) from device-info-related protocol packets and welcome pages of Wi-Fi devices. Using the specific model information of the device, we can learn its current security status. In the security platforms on the Internet, such as CVE [60] and NVD [57], the vulnerability information of the device is well organized and maintained [37]. Therefore, using the device label to retrieve the vulnerability database, we can detect the known vulnerabilities of the device, thereby discovering the vulnerable device and applying further protective measures (e.g., applying security patches to the device or deploying an intrusion detection system [50, 61]).

5. Evaluation

In this section, we evaluate the effectiveness of WND-Identifier through experiments. We first introduce the setup of experimental devices and the collection of data set and then show the results of the evaluation. We enumerate some schemes for device identification and compare them with our solution. Finally, we discuss some issues about WND-Identifier.

5.1. Experiment Settings. To evaluate the effectiveness of WND-Identifier, we choose common wireless devices on the market to conduct experiments. We use a PC running WND-Identifier as the monitoring device to connect to the WLAN. During this period, WND-Identifier utilizes Tcpdump [53] to acquire data packets, processes, and analyzes them to realize the device identification process. In order to be consistent with the real application environment, we set up two types of application scenarios: closed world and open world to simulate the home network and public Wi-Fi environment, respectively.

5.1.1. Closed World. In the closed-world scenario, we have management authority over the wireless network, and all connected devices are known and controlled. Figure 7 shows a test case of the typical home network scenario, in which we set up a home wireless router and connect a group of portable and IoT devices to the Wi-Fi network, such as smartphones, laptops, wireless cameras, sockets, and smart bulbs. We set up three sets of experiments in the closed world with a total of 30 wireless devices from 14 different vendors in 10 categories. Table 3 gives a list of some devices used in our experiments.

5.1.2. Open World. The closed world scene may have contingency and particularity. To show that our solution is highly usable in a broader scene, we set up an open-world scenario to simulate the situation of public Wi-Fi. In this setting, we do not have the management authority over the network but connect to the public Wi-Fi as an ordinary user. Thus, other users and connected devices in the network are unknown and uncontrollable. It is worth noting that we cannot accurately know the number of the devices because some devices may not generate any traffic during our

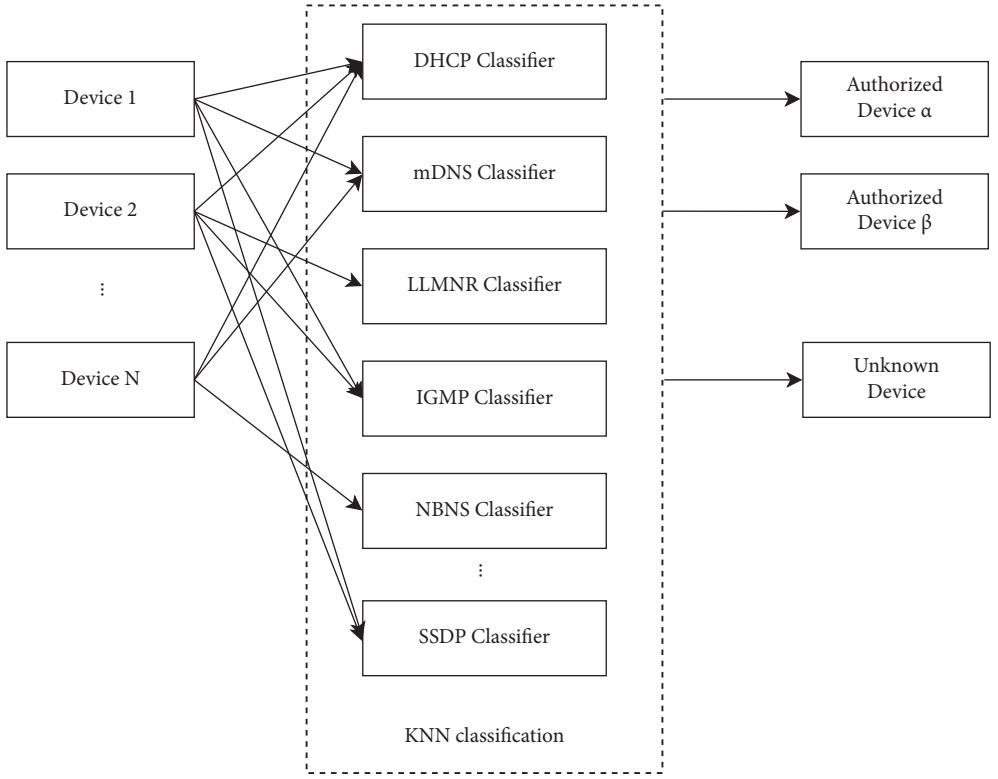


Figure 6: Example of device classification.

Figure 7: A test case of the closed-world scenario.

collection process. Therefore, we take the devices that are obtained the network traffic and MAC address as the experimental data set. In the open-world experiment, we capture data packets of 152 network devices from 18 different vendors in 11 categories.

5.2. Experiment Results. We use the above two types of data sets: open world and closed world for experimental evaluation. In the closed-world scenario, we know the detailed information of the equipment, including (vendor, type, product model) and can directly confirm the accuracy of the device identification. For open-world data sets, we cannot acquire accurate information about the device directly. We label the data set manually and confirm the validity of device labels based on existing information on the Internet, including the official website of the product supplier. For the performance of rule-based device identification and text classification-based identification, we define the following three indicators for evaluation:

Table 3: List of some devices used in our experiment.

Type	Vendor	Model
Router	TP-Link	TL-WR886N
	ASUS	RT-N65R
	Linksys	WRT54G
Smartphone	Apple	iPhone12
	Huawei	P40
	Oppo	A52
TV	Sony	X9500H
Laptop	Apple	Macbook Air
	Huawei	Matebook
IP camera	TP-Link	TL-IPC42A-4
	Ezviz	C6C
	Xiaomi	MJSXJ02CM
Light	Philips	Hue
Kettle	Midea	HE1508a
Socket	LifeSmart	LS060
Gateway	Xiaomi	Gateway-v3

$$C = \frac{\text{labeled devices}}{\text{total devices}}, \tag{3}$$

$$\begin{aligned} A_t &= \frac{\text{correctly mapped devices}}{\text{total devices}}, \\ A_r &= \frac{\text{correctly labeled devices}}{\text{total devices}}, \end{aligned} \tag{4}$$

where C means the coverage rate of extraction rules, A_r represents the accuracy of rule-based device identification (i.e., the correct device labels are generated), and A_t represents the accuracy of text classification-based device identification (i.e., the devices are correctly mapped to authorized or unknown labels).

5.2.1. Performance on Closed-World Datasets. For the data set we collected, we first evaluate the performance of rule-based device identification at three granularity levels (vendor, type, product model). As shown in Figure 8(a), our extraction algorithm achieves $C = 100\%$ at the vendor level in all 30 wireless network devices, and the accuracy A_r reaches 93% and 87% at the device type and product model level, respectively. In the experiment of text classification-based device identification, we first randomly select a dataset of k devices as a benchmark, i.e., the whitelist of legal devices maintained by WND-Identifier. Then, we recollect data three times during the different time periods and connect the remaining devices to the network, i.e., the newly-connected unauthorized devices, to evaluate the validity of text classification-based device identification. It is worth noting that the number of devices in the whitelist has almost no effect on our device identification process. Therefore, we only give the evaluation result of $k = 15$, and the selection of these devices is completely random. We repeat the experiment and achieve 93% accuracy in a total of 85 times of text classification-based device identification.

5.2.2. Performance on Open-World Datasets. For the data set collected from public Wi-Fi, our extraction rules generate 147 valid vendor labels and obtain 136 and 127 for device type and product model, respectively. From this, we can infer the success rate of rule-based device identification, as shown in Figure 8(b). For further text classification-based device identification, we collect data three times under the same Wi-Fi with a sampling time of five minutes. Under the uncontrollable setting of the open world, the access and disconnection of user equipment are completely random. We conduct experiments on two adjacent data sets each time, where the first one is used as the benchmark data set and the latter contains the newly connected devices for identification. As a result, we achieve 85.4% accuracy in 343 times of text classification-based identification.

5.2.3. Performance of Text Classification-Based Device Identification. The text classification model is greatly affected by the size of the training set, so we further explore the impact of sampling duration on device identification. We conduct experiments in the closed-world setting. In the home network environment, we connect 20 devices to the WLAN and evaluate the performance of device identification at different data acquisition duration. As shown in Figure 8(c), only 5 minutes of data collection can achieve 80% accuracy, and after 10 minutes, A_t increases to 95%.

5.3. Comparison. WND-Identifier provides a high-precision and scalable solution for the identification of Wi-Fi devices. In this section, we enumerate relevant researches on device identification in the literature and compare them with our scheme. As shown in Table 4, due to the large differences in the research objects, application scenarios, and identification granularity of each solution, we mainly compare and analyze qualitatively.

5.3.1. Rule-Based Device Identification. Rule-based identification extracts information inherent to the equipment and hard-coded by the manufacturer and directly generates concrete detailed labels for the network device. ARE [35] generates rules for automatically annotating IoT devices, and extracts text information from response data in the application layer. However, active scanning behavior can be easily discovered and blocked by firewall rules, and the application layer response is not always complete, resulting in a limited identification rate at the granularity level of product models. WNV-Detector [37] applies access and extraction rules to obtain fine-grained device information through the management website. However, the scheme requires administrator privileges on the network and is designed for a single type of devices with limited versatility. Moreover, the above schemes extract textual information from just one source, which can be easy for adversaries to modify and deceive. By contrast, our work makes full use of multiple device-info-related protocols and sources of information and combines with further text classification based identification technology to differentiate malicious devices.

5.3.2. Classification Model-Based Device Identification. Device identification based on traffic features actively establishes a connection with the target device and collects response data or passively monitors and acquires network traffic, extracts effective features in the data packets to trains the classification model, and maps the connected device to the correct label. AUDI [26] deduces statistical information based on periodic background traffic to construct device-type fingerprints. GTID [27] uses statistical techniques to capture time-varying behavior of network traffic and identifies devices and types based on ANN. WDMTI [25] relies on the characteristics of DHCP packets when the device is connected and applies the HDP model to classify the device as a known manufacturer/type label or unknown device. IoT Sentinel [30] extracts 23 features from passively observed network traffic and identifies device types based on random forest classifiers. We verify the effectiveness of WND-Identifier on the public dataset [30]. In the identification of all 27 devices, WND-Identifier achieves 100%,

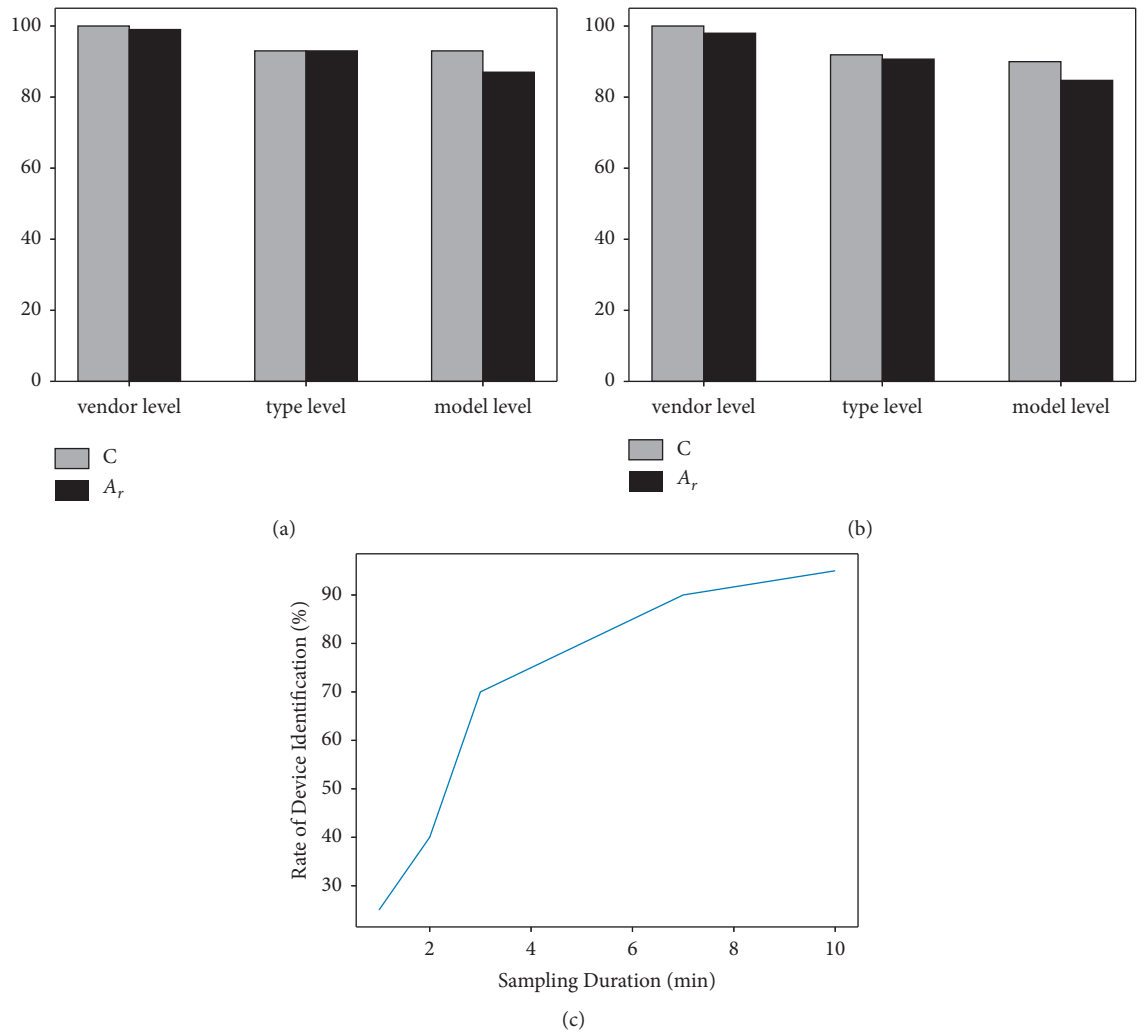


Figure 8: Performance of WND-Identifier. (a) Rule-based identification in the closed-world dataset. (b) Rule-based identification in the open-world dataset. (c) Performance under different sampling durations.

Table 4: Comparison of similar schemes.

Proposal	Object	Rule-based	Model-based	Passive	Concrete label	Output result
WND-Identifier	Wireless device	✓	✓	✓	✓	Vendor, type, model
ARE [35]	Online IoT device	✓			✓	Vendor, type, model
AUDI [26]	IoT device		✓	✓		Type
GTID [27]	Wireless device		✓	✓		Type, model
WDMTI [25]	Wireless device		✓	✓		Vendor, type
IoT Sentinel [30]	Wireless device		✓	✓		Vendor, type
WNV-Detector [37]	Wireless router	✓			✓	Model, version

92.6%, and 85.2% accuracy in the three granularities of vendor, device type, and product model, respectively. The results show that our solution is better than WDMTI and IoT Sentinel. Table 5 shows the results of the comparison, in which the performance of WDMTI and IoT Sentinel is publicly given in [25].

Compared with the above solutions, WND-Identifier can generate fine-grained device model labels, which has obvious advantages in the granularity of identification. In addition, the labels generated by machine learning methods are abstract, and concrete device information cannot be obtained. The correspondence between labels and specific categories needs to be manually marked by the administrator in advance, which is inconvenient for the addition and expansion of new devices. In order to overcome this problem, our solution combines the identification results with the concrete labels generated during the rule-based identification stage to realize an automated, highly available, and scalable identification process.

Table 5: Comparison of performance on public datasets.

Proposal	Accuracy
IoT Sentinel [30]	Type: 81.5%
WDMTI [25] + IoT Sentinel features	Type: 90.1%
WND-Identifier	Vendor: 100%; type: 92.6%; model: 85.2%

5.4. Discussion. WND-Identifier utilizes device-info-related network data packets and key fields to realize automated identification of Wi-Fi devices. Here, we openly discuss some issues about WND-Identifier, including incomplete information, fake device, and original equipment manufacturer.

5.4.1. Incomplete Information. Some devices generate too little traffic, which makes it difficult to extract sufficient device information directly. Instead, we can make full use of limited natural language features, such as keywords like *MSFT*, *Android*, *Homekit*, or special protocols that the device applied, e.g., BROWSER and LLMNR, to identify the type and operating system of the targeted device.

5.4.2. Fake Device. Attackers can use IoT honeypots, emulators, etc., to forge legitimate devices. WND-Identifier can usually utilize the inconsistency in the two-stage device identification process to identify such forgeries. However, it does not rule out that some expert adversaries imitated the network protocol stack perfectly, generating data packets exactly the same as the authorized device. This situation is difficult to distinguish, and the counterfeit device may be identified as legitimate and authentic.

5.4.3. Original Equipment Manufacturer. For some devices, part of their components may be entrusted to other manufacturers, resulting in natural language information from different vendors. In this case, the device identification module may produce ambiguity and obtain contradictory device information. WND-Identifier can identify this inconsistency, require further confirmation by the user, and then add such devices to the whitelist for later identification.

6. Conclusion

The security of wireless networks has received extensive attention. With the rapid growth of wireless devices, more effective strategies are required to classify, manage, and protect them. In this paper, we propose a scheme for automated identification of wireless network devices (WND-Identifier). The technology utilizes device-info-related fields in the data packets and device description information in the welcome interface to generate device labels and maps the text content of the data packets from the connected device that is irrelevant to the concrete application logic to a specific label through text classification based on KNN with TF-IDF, thereby achieving rapid identification of wireless devices. We verify the effectiveness and efficiency of WND-Identifier by conducting experiments on closed and open-world data sets. Our results show that only 10 minutes of data collection can achieve a 95% rate of device identification. Furthermore, we present the expected application scenarios of WND-Identifier, which leverages the inconsistency of identification to identify illegal devices and scans vulnerable devices with the concrete device labels, showing that our solution has a wide range of application prospects.

In the future, we plan to extend WND-Identifier to more protocols (e.g., Bluetooth and Zigbee) and device types. Meanwhile, we will continue to study the identification of hidden devices in the network and propose efficient solutions to protect vulnerable devices. Our work attempts to inspire further research on the security of wireless networks and devices.

Data Availability

The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under grant no. U20B2050 and the Science and Technology Funds from National State Grid Ltd. (The Research on Key Technologies of Distributed Parallel Database Storage and Processing based on Big Data).

References

[1] E. J. Oughton, W. Lehr, K. Katsaros, I. Selinis, D. Bubley, and J. Kusuma, "Revisiting wireless internet connectivity: 5g vs wi-fi 6," *Telecommunications Policy*, vol. 45, no. 5, Article ID 102127, 2021.

[2] "IEEE 802.11ax: The sixth generation of wi-fi white paper," 2021, https://www.cisco.com/c/en/us/products/collateral/wireless/white-paper-c11-740788.html.

[3] "Wi-fi market global forecast," 2021, https://www.marketsandmarkets.com/Market-Reports/global-wi-fi-market-994.html.

[4] M. Vanhoef and F. Piessens, "Key reinstallation attacks: forcing nonce reuse in wpa2," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1313–1328, Dallas Texas, USA, October 2017.

[5] "Kr00k–CVE-2019-15126: A serious vulnerability deep inside wi-fi encryption," 2021, https://www.welivesecurity.com/wp-content/uploads/2020/02/ESETKr00k.pdf.

[6] Y. Meidan, M. Bohadana, A. Shabtai et al., "Detection of unauthorized iot devices using machine learning techniques," 2017, https://arxiv.org/abs/1709.04647.

[7] B. Morrow, "Byod security challenges: control and protect your most sensitive data," *Network Security*, vol. 2012, no. 12, pp. 5–8, Article ID 701113, 2012.

[8] Z. Zhang, H. Hasegawa, Y. Yamaguchi, and H. Shimada, "Rogue wireless ap detection using delay fluctuation in backbone network,"vol. 1, pp. 936-937, in *Proceedings of the 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 1, pp. 936-937, IEEE, Milwaukee, WI, USA, July 2019.

[9] P. Auffret, "Sinfp, unification of active and passive operating system fingerprinting," *Journal in Computer Virology*, vol. 6, no. 3, pp. 197–205, 2010.

[10] Y.-C. Chen, Y. Liao, M. Baldi, S.-J. Lee, and L. Qiu, "Os fingerprinting and tethering detection in mobile networks," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, pp. 173–180, Vancouver, Canada, November 2014.

[11] Z. Shamsi, A. Nandwani, D. Leonard, and D. Loguinov, "Hershel: single-packet os fingerprinting," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2196–2209, 2015.

[12] Z. Shamsi, D. B. Cline, and D. Loguinov, "A non-parametric iterative classifier for internet-wide os fingerprinting," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 971–982, Dallas, TX, USA, November 2017.

[13] P. Eckersley, "How unique is your web browser?" in *Proceedings of the International Symposium on Privacy Enhancing Technologies Symposium*, pp. 1–18, Springer, Berlin, Germany, July 2010.

[14] A. Gómez-Boix, P. Laperdrix, and B. Baudry, "Hiding in the crowd: an analysis of the effectiveness of browser fingerprinting at large scale," in *Proceedings of the 2018 world wide web conference*, pp. 309–318, Lyon, France, April 2018.

[15] A. Vastel, P. Laperdrix, W. Rudametkin, and R. Rouvoy, "Fp-scanner: the privacy implications of browser fingerprint inconsistencies," in *Proceedings of the 27th USENIX Security Symposium (USENIX Security 18)*, pp. 135–150, Baltimore, MD, USA, Auguest 2018.

[16] O. Starov and N. Nikiforakis, "Xhound: quantifying the fingerprintability of browser extensions," in *Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP)*, pp. 941–956, IEEE, San jose, CA, USA, May 2017.

[17] K.-F. Kao, T.-H. Yeo, W.-S. Yong, and H.-H. Chen, "A location-aware rogue ap detection system based on wireless packet sniffing of sensor APS," in *Proceedings of the 2011 ACM Symposium on Applied Computing*, pp. 32–36, TaiChung, Taiwan, March 2011.

[18] H. Han, B. Sheng, C. C. Tan, Q. Li, and S. Lu, "A timing-based scheme for rogue ap detection," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 11, pp. 1912–1925, 2011.

[19] V. S. Sriram, G. Sahoo, and K. K. Agrawal, "Detecting and eliminating rogue access points in IEEE-802.11 WLAN-a multi-agent sourcing methodology," in *Proceedings of the 2010 IEEE 2nd International Advance Computing Conference (IACC)*, pp. 256–260, IEEE, Patiala, India, February 2010.

[20] W. Wu, X. Gu, K. Dong, X. Shi, and M. Yang, "A novel received signal strength–based approach for practical rogue access point detection," *International Journal of Distributed Sensor Networks*, vol. 14, no. 8, Article ID 1550147718795838, 2018.

[21] G. F. Lyon, *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*, Insecure. Com LLC (US), SeattleWA, USA, 2008.

[22] S. Bratus, C. Cornelius, D. Kotz, and D. Peebles, "Active behavioral fingerprinting of wireless devices," in *Proceedings of the first ACM conference on Wireless network security*, pp. 56–61, Alexandria, VA, USA, April 2008.

[23] K. Gao, C. Corbett, and R. Beyah, ""A passive approach to wireless device fingerprinting," in *Proceedings of the IEEE/IFIP International Conference on Dependable Systems & Networks (DSN)*, pp. 383–392, IEEE, Chicago, IL, USA, July 2010.

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

[25] L. Yu, T. Liu, Z. Zhou, Y. Zhu, Q. Liu, and J. Tan, "Wireless device manufacturer and type identification using hierarchical Dirichlet process," in *Proceedings of the IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 19–27, IEEE, Chengdu, China, October 2018.

[26] S. Marchal, M. Miettinen, T. D. Nguyen, A.-R. Sadeghi, and N. Asokan, "AuDI: toward autonomous IoT device-type identification using periodic communication," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1402–1412, 2019.

[27] S. V. Radhakrishnan, A. S. Uluagac, and R. Beyah, "A technique for physical device and device type fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, vol. 12, no. 5, pp. 519–532, 2014.

[28] A. Sivanathan, D. Sherratt, H. H. Gharakheili et al., "Characterizing and classifying iot traffic in smart cities and campuses," ", IEEE, in *Proceedings of the 2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 559–564, November 2017.

[29] L. Yu, B. Luo, J. Ma, Z. Zhou, and Q. Liu, "You are what you broadcast: identification of mobile and IoT devices from (public) wifi," in *Proceedings of the 29th USENIX Security Symposium (USENIX Security 20)*, pp. 55–72, Boston, MA, USA, Auguest 2020.

[30] 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 of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pp. 2177–2184, IEEE, Atlanta, GA, USA, June 2017.

[31] Masscan, "Mass IP port scanner," 2021, https://github.com/robertdavidgraham/masscan.

[32] Zmap, "A fast single-packet network scanner," 2021, https://zmap.io/.

[33] J. Matherly, *Complete Guide to Shodan*, Vol. 1, Shodan, LLC, , Pflugerville, TX, US, 2015.

[34] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman, "A search engine backed by internet-wide scanning," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 542–553, Denver, CO, USA, October 2015.

[35] X. Feng, Q. Li, H. Wang, and L. Sun, "Acquisitional rule-based engine for discovering internet-of-things devices," in *Proceedings of the 27th USENIX Security Symposium (USENIX Security 18)*, pp. 327–341, BALTIMORE, MD, USA, Auguest 2018.

[36] K. Yang, Q. Li, and L. Sun, "Towards automatic fingerprinting of iot devices in the cyberspace," *Computer Networks*, vol. 148, pp. 318–327, 2019.

[37] Y. Huang, F. Zhu, L. Liu et al., "Wnv-detector: automated and scalable detection of wireless network vulnerabilities,"

EURASIP Journal on Wireless Communications and Networking, vol. 2021, no. 1, pp. 1–21, Article ID 019784, 2021.

[38] F. Zhu, L. Liu, W. Meng, T. Lv, S. Hu, and R. Ye, "Scaffisd: a scalable framework for fine-grained identification and security detection of wireless routers," in *Proceedings of the 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 1194–1199, IEEE, Guangzhou, China, Decmber 2020.

[39] J. Caballero, S. Venkataraman, P. Poosankam, M. G. Kang, D. Song, and A. Blum, "Fig: automatic fingerprint generation," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2007*, San Diego, CA, USA, March 2007.

[40] Q. Xu, R. Zheng, W. Saad, and Z. Han, "Device fingerprinting in wireless networks: challenges and opportunities," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 94–104, 2015.

[41] I. Sanchez-Rola, I. Santos, and D. Balzarotti, "Clock around the clock: time-based device fingerprinting," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1502–1514, Toronto, Canada, October 2018.

[42] A. C. Polak, S. Dolatshahi, and D. L. Goeckel, "Identifying wireless users via transmitter imperfections," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 7, pp. 1469–1479, 2011.

[43] S. U. Rehman, K. W. Sowerby, and C. Coghill, "Analysis of impersonation attacks on systems using rf fingerprinting and low-end receivers," *Journal of Computer and System Sciences*, vol. 80, no. 3, pp. 591–601, 2014.

[44] C. Shen, C. Liu, H. Tan, Z. Wang, D. Xu, and X. Su, "Hybrid-augmented device fingerprinting for intrusion detection in industrial control system networks," *IEEE Wireless Communications*, vol. 25, no. 6, pp. 26–31, 2018.

[45] T. S. Kondo and L. J. Mselle, "Penetration testing with banner grabbers and packet sniffers," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 5, no. 4, pp. 321–327, 2014.

[46] S. Lee, S.-H. Shin, and B.-h. Roh, "Abnormal behavior-based detection of shodan and CENSYS-like scanning," in *Proceedings of the 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 1048–1052, IEEE, Milan, Italy, July 2017.

[47] R. Mitev, A. Pazii, M. Miettinen, W. Enck, and A.-R. Sadeghi, "Leakypick: iot audio spy detector," in *Proceedings of the Annual Computer Security Applications Conference*, pp. 694–705, Austin, USA, December 2020.

[48] E. López-Morales, C. Rubio-Medrano, A. Doupé et al., "Honeyplc: a next-generation honeypot for industrial control systems," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pp. 279–291, Virtual Event. USA, November 2020.

[49] H. Šemić and S. Mrdovic, "Iot honeypot: a multi-component solution for handling manual and mirai-based attacks," in *Proceedings of the 2017 25th Telecommunication Forum (TELFOR)*, pp. 1–4, IEEE, Belgrade, Serbia, November 2017.

[50] X. Feng, X. Liao, X. Wang et al., "Understanding and securing device vulnerabilities through automated bug report analysis," in *Proceedings of the SEC'19: 28th USENIX Conference on Security Symposium*, Santa Clara CA USA, Auguest 2019.

[51] A. Pandey and J. R. Saini, "Counter measures to combat misuses of mac address spoofing techniques," *International Journal of Advanced Networking and Applications*, vol. 3, no. 5, p. 1358, 2012.

[52] J. Yu, E. Kim, H. Kim, and J. Huh, "A framework for detecting mac and ip spoofing attacks with network characteristics," in *Proceedings of the 2016 International Conference on Software Security and Assurance (ICSSA)*, pp. 49–53, IEEE, St. Pölten, Austria, Auguest 2016.

[53] Tcpdump, "A powerful command-line packet analyzer," 2021, https://www.tcpdump.org/.

[54] Wireshark, "A widely-used network protocol analyzer," 2021, https://www.wireshark.org/.

[55] G. Joshua Wright and C. Joshua, *Detecting Wireless Lan Mac Address Spoofing*, Cisco Certified Network Associate, Silicon Valley, CA, USA, 2003.

[56] B. Alotaibi and K. Elleithy, "A new mac address spoofing detection technique based on random forests," *Sensors*, vol. 16, no. 3, p. 281, 2016.

[57] *National vulnerability database*, https://nvd.nist.gov, 2021.

[58] S. Qaiser and R. Ali, "Text mining: use of TF-IDF to examine the relevance of words to documents," *International Journal of Computer Application*, vol. 181, no. 1, pp. 25–29, 2018.

[59] P. Lu, "A position self-adaptive method to detect fake access points," *Journal of Quantum Computing*, vol. 2, no. 2, pp. 119–127, 2020.

[60] "Common vulnerabilities and exposures," 2021, http://cve.mitre.org.

[61] T. D. Nguyen, S. Marchal, M. Miettinen, M. Miettinen, N. Asokan, and A. Sadeghi, "D'iot: a federated self-learning anomaly detection system for iot," in *Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 756–767, IEEE, Dallas, TX, USA, July 2019.