Cross-Layer Protocol Fingerprint for Large-Scale Fine-Grain Devices Identification

DAN YU, HAOGUANG XIN, YONGLE CHEN, (Member, IEEE), YAO MA, AND JUNJIE CHEN
College of Information and Computer, Taiyuan University of Technology, Taiyuan 030024, China

Corresponding author: Yongle Chen (chenyongle@tyut.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB0803402, in part by the Natural Science Foundation of Shanxi Province under Grant 201701D111002, and in part by the Key Research and Development Program of Shanxi Province under Grant 201903D121121.

ABSTRACT Internet-connected Internet of Things (IoT) devices are exploding, which pose a significant threat for their management and security protection. IoT device identification is a prerequisite for discovering, monitoring, and protecting these devices. Although the existing proactive identification methods based on protocol fingerprint can discover and identify large-scale IoT devices, the fingerprint granularity is difficult to meet the requirements of security risk assessment for large-scale IoT devices. Since IoT devices usually support multiple network protocols for specific collection and control tasks, we propose a cross-layer protocol fingerprint to achieve large-scale fine-grained devices identification instead of traditional single protocol fingerprint. We first design a probing scheme for gathering HTTP and TCP cross-layer packets. Then we select the specific field of the HTTP and TCP protocols based on the diversity and consistence of field value. Finally, we utilize convolutional neural network (CNN) and long-term memory network (LSTM) to extract and construct feature fingerprint of these specific fields, and achieve a fine-grain IoT devices identification with high accuracy. The experimental results show that our identification accuracy of devices model reaches 96.6%, the recall rate reaches 97.4%.

INDEX TERMS Internet of Things, devices identification, cross-layer, fine-grain, neural network.

I. INTRODUCTION

With Internet of Things (IoT) devices exploding, the number of Internet-connected devices will grow rapidly and its impact on society is also gradually increasing. By 2021, the spending on the IoT device market will grow to US$520 billion per year [1]. At the same time, more and more commercial Internet of things applications have been developed, such as the mobile crowdsourcing mentioned by Zhibo Wang et al, which is becoming an indispensable part of our life [2], which highlights the importance of Internet of things devices. IoT devices are the source of perceiving data and information, and they are also the most vulnerable part on Internet. Due to the memory and computing power of IoT devices is limited, they are more susceptible to cyber attacks, destruction and reverse engineering than traditional computers. Most IoT devices have security risks such as information leakage, lack of security protection, and hysteretic firmware upgrades, which pose great security threats to the entire network environment.

Meanwhile, the imperfect security management in the IoT applications further increases the risk of devices being attacked. Privacy leakage is particularly serious, and many security researchers have put forward their own methods and opinions. For instance, Mengkai Song et al. analyzed the problem of users’ privacy being leaked due to malicious attacks on servers [3]. Vulnerability analysis of IoT devices is a prerequisite for reducing IoT security threats, and IoT devices identification technology is the predominate method of IoT devices vulnerability analysis.

The existing IoT devices identification technology mainly uses the protocol supported by the device to obtain the communication traffic between the devices in a proactive or passive way, and then analyze the packets content characteristics through statistics or machine learning to achieve the device identification. Passive IoT devices identification methods require the deployment of traffic detection software in network routes. However, it is difficult to deploy traffic detection software on the backbone Internet containing a large number of IoT devices, which leads to the lack of communication traffic data, and cannot achieve the accurate...
devices identification in the large-scale Internet. The proactive identification method can send a request packet to any devices, and identify the device attribute (e.g., type, brand, model) according to the diversity of response. In different protocols, the field values in the response packet headers are different, which can be used for devices identification [4]. The banner-based identification method based on application protocol is a predominant proactive technology since the packet content in the application protocol banners usually contains device attributes [5]. The well-known Shodan search engine adopts banner-based method to achieve identification accuracy of 95% [6]. The identification is accurate, but the recall rate of device attribute information is low, resulting in lower identification granularity. Another popular proactive method is fingerprint-based device identification methods. Nmap is widely used by network security researchers for device identification and system detection. At the beginning of the identification process, Nmap sends 15 groups of detection packets to the target device, and builds device fingerprints based on the field characteristics of the response packets from TCP/IP and ICMP to complete devices identification, such as Win, TTL. For limited number of operating systems, the identification accuracy is acceptable, while for the massive IoT devices, the identification accuracy will decrease significantly.

In this paper, we propose a fine-grained large-scale proactive IoT devices identification method based on cross-layer protocol fingerprint. We find most IoT devices currently support HTTP protocol for connecting Internet. The response packet of HTTP protocol contains a large number of device banner information. Furthermore, HTTP protocol utilize TCP to transmit web data, and different devices return different TCP field values. By simulating the real device communication process, we take the initiative to establish a connection with the target device and send conventional HTTP request packets, so as to obtain the HTTP and TCP protocol response packets from the target device. We design the cross-layer fingerprint based on HTTP and TCP protocol. We select the response packet fields based on the consistence and diversity metrics in our statistical experiments. Among them, we select 4 kinds of fields in the HTTP response packet and 2 kinds of fields in the TCP response packet. Then we use these fields as the input of the CNN+LSTM neural network model to further complete the feature extraction and fingerprint construction. Finally, we combine the SoftMax function to obtain the probability distribution of the device type, brand, and model. We can realize the accuracy devices identification based on application layer and transport layer.

Our main contributions are described as following:

- We design a simple cross-layer protocol probing scheme, which can obtain the HTTP and TCP protocol fields by sending single regular HTTP requests to the target device.
- We propose a devices fingerprint generation method based on HTTP and TCP protocols field. The consistence and diversity metric are designed for selecting the feature fields.
- We implement the device identification prototype system by CNN+LSTM+SoftMax neural network model. We use the neural network model to extract feature and construct fingerprint, and realized the large-scale fine-grain devices identification. The experimental results show that our identification accuracy of devices model reaches 96.6%, the recall rate reaches 97.4%.

The rest of this paper is structured as follows. Section 2 introduces the work related. Section 3 describes the devices identification framework. Section 4 describes in detail our devices fingerprint. Section 5 shows the fine-grain devices identification method. Section 6 shows the experimental evaluation. Section 7 summarizes our work.

II. RELATED WORK

With the emergence of cyberspace security issues, cybersecurity researchers have also made further developments in the identification technology of IoT devices. The existing devices identification methods are mainly classified into two types, one is passive identification, the other is proactive identification.

In the passive identification methods, various characteristics of communication between devices can be used to realize devices identification. In the field of wireless devices, in view of the security challenges of network intrusion detection and prevention in the decentralized architecture of ZigBee Ad-Hoc network, the unique original attribute of radio frequency between devices (RF-DNA) can be used as fingerprint to realize identification [7]. Kohno et al. proposed a method of fingerprinting the device using the clock offset related to the device hardware [8]. Meidan et al. analyzed network traffic and used machine learning algorithms for device identification for the first time [9]. The method proposed by Caballero et al. can use detection packets and combine machine learning methods to generate device fingerprints for device identification [10]. Brik et al. proposed that the features that can be used in the MAC layer are relatively easy to extract and do not require special hardware. Different vendors implement different details that are not specified in the MAC layer standard, resulting in different devices may have different features and realize device identification method [11]. Radhakrishnan et al. proposed a device fingerprint technology named GTID, which uses the information leaked by heterogeneous devices and applies it to the network traffic generated by the device to generate the device fingerprint [12]. Because the architectures of firmware (firmware, driver, etc.) produced by different manufacturers are heterogeneous, resulting in different packet interarrival time, Gao et al. used these packet sequence characteristics as fingerprints to determine the type of access point (AP) [13]. Miettinen et al. proposed the concept of IoT SENTINEL, which generates device fingerprints by capturing the behavior of the device during the setting process, and realized a passive
devices identification method, but its identification granularity can only reach the device type [14]. In the traffic generated by the device, the length of the data packet is different. This feature can also be used for device identification. Antonio et al. used the length of the data packet in the flow generated by the device as a fingerprint to describe the behavior of the device or event, although the generated network traffic load small, but the premise is that the traffic generated by the IoT device to be identified must have a packet length pattern used to describe the characteristics of the data source [15].

Compared with the passive identification method that collects user traffic in the network, the proactive identification method can realize device identification on the basis of protecting user privacy by sending detection packets to the device under test. In the proactive identification method, the well-known Nmap software sends a large number of detection packets to the device under test, which leads to a large amount of time overhead in the identification process, which occupies network bandwidth and reduces network communication efficiency. Bratus et al. proposed a method of actively sending non-standard or incorrectly formatted 802.11 frames to observe the response of the device to fingerprint the device. Since the response of the counterfeit device may be very different from the original device, this is good to identify the devices that have changed their MAC addresses [16]. In the existing cross-layer identification method, Yang et al. extracted the features of the 24 communication protocols in the application layer, transport layer, and network layer of the device for device identification, and realized multiple device identification including industrial control equipment. Owing to the use of a large number of protocol features, it cannot be used well in large-scale network identification [17]. The current proactive identification method needs to send a large number of detection packets, which undoubtedly increases the network bandwidth occupation and increases the time for device identification. Our method only utilizes two kinds of protocol communication packets, and realizes high-accuracy device identification under the condition of occupying a small amount of network resources.

III. DEVICES IDENTIFICATION FRAMEWORK

In order to achieve fine-grained device identification in large-scale networks, we need to use a lightweight method of capturing device characteristics that are differentiated and stable enough among similar types of devices. Based on this idea, we propose a device identification method based on cross-layer protocol. We simulate the communication process between devices in a real network, actively establish a connection to the target device and send an HTTP request, obtains communication packets from the target device. Then we use the selected fields in these packets to realize devices identification in large-scale networks by neural network model. As shown in Figure 1, our cross-layer devices identification method is divided into three phases: 1. Data collection. First, we construct a request data packet and send it to the target device. After receiving the request, the target device returns HTTP and TCP response packets. Finally, we use the packet sniffer to capture these communication packets. 2. Fingerprint generation. The selected field values are embedded into the vector in binary form and converted into a feature matrix to generate the device fingerprint. 3. Device identification. We use the neural network model to further extract the features, and obtain the probability distribution of the device to complete the device identification process.

IV. DATA COLLECTION AND DEVICE FINGERPRINT

In existing device identification methods, identification methods based on banner of application layer protocol are predominate [18]. IoT devices will use different types of protocols to implement diverse functions, such as HTTP, Telnet, DNS, UPnP, etc. Different device types have large differences in the value of some fields in the protocol. For example, for a network camera that transmits image data, the value of the corresponding packet length in the transmission data will be much larger than that of other typical network devices.

A. PROTOCOLS SELECTION

The well-known search engine Shodan [6] and ZoomEye [19] realize the identification of IoT devices through multiple application layer protocols. They are the typical banner-based devices identification method whose identification accuracy is high. But it needs to implement a large number of complex protocol detection processes, which requires more time cost and network bandwidth. Since the identification of the application layer protocol contains a variety of fields describing device information, it can play an important role
in devices identification. We also use several application layer protocol contents as features of device identification. Meanwhile, when network devices transmit data through the transport layer protocols, they can be further used to achieve fine-grained devices identification due to their different characteristics such as the type and size of the data they transmit. In order to decrease the time consumption and network bandwidth occupation as much as possible without reducing the identification accuracy, the packet content of the selected protocols must achieve the following three requirements: 1. The field values in selected protocol keep consistent in the same kind of device. 2. The field values in selected protocol is diverse in different kinds of devices. 3. In order to reduce overhead, we adopt fewer types of protocols which are predominate in actual device applications and contain rich features to distinguish devices. In this paper, we implement fine-grained devices identification for the three types of devices: network cameras, routers, and printers. We have counted the popular protocols for network communication between these three types of devices. The solid circle indicates that the device supports this protocol, and the ring indicates that the one part of devices supports the protocol, and the hollow circle indicates that the device does not support the protocol.

For the device-support protocols in Table 1, we find that some of them must be manually enabled in the device before using, especially the UPnP protocol. Although most IoT devices support this UPnP protocol, it is real used only in routers. Furthermore, UPnP needs to be manually turned on and brings more security risks. Especially in some network scenarios with high security requirements, its practical application is subject to some restrictions [20]. Through experimental verification, we find there are some detail information that can describe the device type in the HTTP packet. For instance, the Server field content indicates the server software information that processes the HTTP request, and it contains the annotation information of the software. Content-Type is used to identify the type of entity to the request body. Content-Length refers to the size of the entity. Due to the different functional configurations and implementation of different devices manufacturers, there are differences in the transmission content of the HTTP protocol used by different devices. Furthermore, we can use the diversity between these information as the features of the target devices, and then combine the machine learning model to identify the devices.

In the proactive identification process, we known Nmap needs to send a large number of packets to the target device, and uses the TTL, WIN and other fields in the various protocol packets response from the target device as the features of device identification. However, the probing packets sent by Nmap are possibly to be rejected or discarded by the target device owing to over probing. This also results in that Nmap cannot be well applied in a large number of IoT devices. The HTTP specifies the format of data packets transmitted on the network, but the detail data transmission between real network devices is implemented by TCP/IP. Therefore, we finally adopt HTTP and TCP protocol content as the feature for device identification.

### B. PROTOCOL FIELD SELECTION

HTTP and TCP protocols contain a variety of fields, and the meanings of the fields are different. We cannot determine which fields in the protocol can be used as the features for device identification. We extract the field values in HTTP and TCP communication packets from a large number of devices to analyze and compare the fields contents, and filter out fields that are diverse in different devices. In this section, we take a printer (printer brand model: HP 5740) as an example to illustrate how we filter HTTP and TCP packet fields.

The Server field describes the software information about the device. This field usually contains the specific model information of the device. The information we extracted from the HTTP response packet of the HP 5740 printer is following: HP Officejet 5740 series-B9S76A; Serial Number: TH4AD3303V05ZF. We can find that the detail model of the device has been clearly indicated in this field. The Content-type field indicates the MIME type of the resource. Different devices inform the request body type of content actually returned according to the characteristics of the resource transmitted by itself. In this example, the value is text/html. The value of other different devices may be text/plain, image/png and so on. The Cache-control field specifies the caching mechanism that the request and response should follow. The value of the modified field in this example is “max-age=0”. Different devices indicate whether the response can be cached according to their own conditions, or neither request nor response information uses cache to prevent information leakage. The above three fields are closely related to the characteristics of the device itself. Although we consider above fields suitable as the device fingerprint, we still adopt the experimental and mathematical methods to verify them.

Table 2 and Table 3 list the real values of some fields that we have collected and calculate from different devices HTTP and TCP protocol packets. In this experiment, we send regular HTTP requests to a large number of devices. The

---

**TABLE 1.** Predominate protocols in three type of devices.

| Protocol | Network camera | Router | Printer |
|----------|----------------|--------|---------|
| UPnP     | ○              | ○      | ○       |
| HTTP     | ●              | ●      | ●       |
| FTP      | ○              | ●      | ○       |
| Telnet   | ●              | ●      | ○       |
| RTSP     | ●              | ○      | ○       |
| SSH      | ○              | ●      | ○       |
| SNMP     | ○              | ○      | ○       |
| TCP      | ●              | ●      | ●       |
| UDP      | ●              | ●      | ●       |
Maximum Segment Size (MSS) determines the maximum data segment allowed to be transmitted in a TCP packet. In the process of each device returning data, if the data size exceeds this value, the data will be split into multiple transmissions. We can obtain multiple TCP Segment values from these communication packets, and then calculate the average of the TCP Segment, short for Ave-TCP Segment. We also obtain the Ave-WIN value by the same way. We compare these field values in different types of devices and similar model of devices. It can be shown from the table that the value of Content-Length in the packet of HP-5740 is 1205 and in the packet of DCS-930LB1 is 490. The TCP Segment field here indicates the size of the data transmitted every time, which keeps better consistency in the same devices and diversity in the different devices.

In order to quantize the consistency and diversity of device features by mathematical method. We first measure the consistency of field values with a uniform standard, we calculated the coefficient of variation of these fields as followed:

$$C_v = \frac{\sigma}{\mu}$$

where $C_v$ represents the coefficient of variation, $\sigma$ means standard deviation, $\mu$ means average value. The smaller the value of $C_v$, the smaller the discreteness of the field value, in particular, the more consistent. Furthermore, by averaging the coefficient of variation of multiple devices, we can get the dispersion degree of the fields. Next, we use Euclidean distance to measure the difference between these fields as followed:

$$Sim(x, y) = \frac{1}{1 + d(x, y)}$$

$$d(x, y) = \sum_{i=1}^{m} \sqrt{x_i^2 - y_i^2}$$

where $d(x, y)$ represents the Euclidean distance between $x$ and $y$, and $x_i$ and $y_i$ represent the value of the field in different devices respectively. We normalize the value of Euclidean distance into $[0, 1]$, and define $Sim$ as the similarity. The smaller the $Sim$, the larger the $d(x, y)$, which means that the diversity between the fields of different devices is larger.

The above table lists the calculation results of the average coefficient of variation $C_v$ and average similarity $Sim'$ of 3000 devices fields. The average coefficient of variation of the three fields in the table is less than 0.1, and the degree of dispersion is small. That is, the field is relatively consistent. Meanwhile, their average similarity is also small, indicating that these fields are quite diverse among different devices. Therefore, the three fields in the table can be used as devices fingerprint. Through a great deal of experiment, protocol fields we finally filtrated are shown in the following table.

C. PACKETS COLLECTION SCHEME

In the process of collecting protocol fields, we find that the value of the same field may have the same value in different devices. Although the proportion of the same value is small, it may also reduce the granularity of device identification. Inspired by Bezawada et al, they use the data extracted from the packets generated by the device as a feature to describe the device behavior [21]. We can utilize the entire data packet generated by the target device as a feature to describe the behavior of the device. As we know, CNN and LSTM network can extract the content and timing features in the data, so we input the packets returned by the target device into the neural network model in order for further feature extraction. In this way, not only the granularity of device identification can be further improved, but also the data returned by the target device can be used efficiently. The rest of this section will explain which packets we use as device fingerprints to describe the device behavior.

The detailed process is in Figure 2. We actively establish a connection with the target device, and then send an HTTP request. After receiving the request, the target device returns TCP response packets to transmits data. Finally, the request client disconnects from the target device. In this process, a total of k+4 response packets were received, including...
FIGURE 2. Data packets collection.

one TCP packet with SYN+ACK flag during the TCP triple handshake phase, k TCP response packets for transmitting data, one HTTP response packet, one TCP package with ACK flags and one TCP package with FIN flag. We utilize five of these packets to describe the device behavior. The first packet is the packet with the SYN flag returned by the target device during the TCP connection establishment phase, and the second is the first packet of transmits data returned by the target device. The reason why only the first packet in the transmission data is used here is that some devices return a small amount of data and only one packet can complete the data transmission task. The third packet is the HTTP response packet with 200 OK. The fourth and fifth packets are the TCP packets with ACK and ACK+FIN flags returned by the target device when TCP disconnects. These packages are represented by red lines in the figure 2. If a redirection packet is generated in the process of sending a request from the target device, we will send the request to the target address again.

It is noteworthy that large-scale scanning of IoT devices will occupy a large amount of network resources, which is immoral. We only scan the public devices in the network, and the scanning frequency of the same device are not more than two times, so it will not pose a threat to the security of these devices.

V. FINE-GRAIN DEVICES IDENTIFICATION

The above section explains the HTTP and TCP protocols data that can be invoked as the source of device identification feature. The data first need to be preprocessed for model training. In this section, we will introduce the details of the device identification process. It mainly divided into three parts, data preprocessing, feature extraction, and devices identification. After extracting the data, we will filter it in the data preprocessing stage, then complete feature extraction and model training. The feature extraction process is implemented using a neural network model (CNN+LSTM). Since the data packets generated by the target device are ordered, that is to say, these data packets not only have differences in content, but also have some time diversity. We utilize the data packets and the filtered fields as the input of the neural network, and then design convolution kernels of convolutional neural networks for feature extraction, and reduce the amount of calculations through pooling layers. The convolution kernel can extract features in the data and obtain a series of time-dependent feature vectors. Multiple convolutional layers can extract deeper features, and then input them into the LSTM network model for encoding. As we all know, LSTM network has excellent performance in dealing with time series problems. Therefore, we input the extracted feature vector into the fully connected neural network layer and classified them using the SoftMax function. The model training process proposed in this paper is carried out in a supervised manner. The process of feature extraction and model training is shown in Figure 3.

FIGURE 3. Neural network model.

A. DATA PREPROCESSING

We extract the communication data packets and field content collected by the packet sniffer, by combining with NLTK library, we delete the stop words, time and other data that are not useful for devices identification in HTTP fields, and remove the useless symbol data in the fields, such as ‘.’, ‘/’. Since the input data of CNN must be a fixed size, we embed each byte of these data packets and field values into an 8-dimensional vector in binary form, and finally combine with zero-padding method to generate the feature matrix which is used as the device fingerprint.

B. FEATURE EXTRACTION

In the actual application process, the convolutional neural network is often composed of multiple convolutional layers, pooling layers and fully connected layers. The convolution
operation is completed by the convolution kernel. The purpose of the convolution operation is to extract the characteristics of the input data, the convolutional layer and the pooling layer are interleaved, and the output of the last convolutional layer is used as the input of the fully connected layer. The role of the pooling layer is to reduce the size of the output data volume of the convolutional layer. By adding a pooling layer between the convolutional layers, we can reduce the amount of calculation and avoid model overfitting. Because each operation in each convolutional layer is linear, we have introduced an activation function, and the output of each convolutional layer is activated by the leaky rectified linear unit (LReLU). Consequently, we can avoid the problem that the output of the neural network is only the linear combination of input. In order to automatically capture the global feature of the data, we flatten the output in the 4th convolutional layer, and dropout with 50% probability is used to avoid overfitting. Finally, the local features extracted in the convolutional layer and the pooling layer are input into two fully connected layers, where each fully connected layer contains 256 neurons, combined into the overall feature and output feature vector v_t.

We make the feature vector v_t output from CNN model as a time-series sequence and input it into LSTM model for encoding. In the process of neural network design and coding, we find that the correlation between the features in HTTP and TCP messages is different, that is, different data features make different contributions to device identification. So we set W_f in the forgetting gate. f_t is the information that needs to be discarded to weaken the influence of some weaker features associated with device identification:

\[ f_t = \sigma (W_f \cdot [h_{t-1}, x_t] + b_f) \]  (3)

where \( \sigma \) is the activation function named sigmoid, the range of the value is [0, 1], when it is 0, it means that all the information is not passed, when it is 1, it means that all the information is passed, W_f is the forgetting weight, h_{t-1} is the state of the upper hidden layer, x_t is the input at this moment. b_f is the offset value of the forget gate.

For some features that are strongly related to devices identification, using the memory gate as the feature that needs to be memorized. At this moment, the value of the memory gate i_t and the instant cell state C_t are output.

\[ i_t = \sigma (W_i \cdot [h_{t-1}, x_t] + b_i) \]  (4)

\[ \tilde{C}_t = tanh (W_C \cdot [h_{t-1}, x_t] + b_C) \]  (5)

The tanh is the activation function, W_i is the memory weight, b_i is the offset value of the memory gate and W_C is the weight matrix of input gate.

We multiply the old cell state by the value of the forget gate, and add the product of the instant cell state multiplied by the memory gate to get the current cell state C_t. In essence, this step is the information update process. After this step, we can add information related to device identification to the new cell state, and remove information unrelated to device identification.

\[ C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \]  (6)

After the cell state is updated, we adopt the sigmoid function to determine which parts in h_{t-1} and x_t to output at this moment, and O_t is the current output:

\[ O_t = \sigma (W_o \cdot [h_{t-1}, x_t] + b_o) \]  (7)

We further use the tanh function to convert the new cell state range from -1 to 1, and then multiply it with the output gate O_t. Finally, we can get the output of the hidden layer at this moment:

\[ h_t = O_t \cdot tanh (C_t) \]  (8)

Through the calculation of the above formula, the LSTM network model extracts features that are useful for devices identification, generates a vector and inputs it into a fully connected neural network with SoftMax function to obtain the probability of the device type, brand or model, and finally completes the identification process.

C. DEVICES IDENTIFICATION

For the feature vector v_t of each device, a vector h_t is generated after LSTM encoding. We input h_t into the fully connected neural network with SoftMax function, and finally calculate the probability P_t to complete the devices identification process. The calculation method of P_t is as follows:

\[ P_t = \frac{\exp (h_t)}{\sum_{i=0}^{n} \exp (h_i)} \]  (9)

where n represents the total number of samples, that is, the number of devices that need to be classified. Through the SoftMax function, according to the output vector of neural network, the classification result is transformed into values ranging from 0 to 1. Each value represents the probability of belonging to a specific device type, and the sum of the probabilities of all outputs is equal to 1.

VI. EXPERIMENTS EVALUATION

In this paper, we implement our cross-layer fine-grained devices identification method on three device types: network cameras, routers, and printers. In order to obtain a large number of devices data for training neural network models, we utilized Censys API [22] for device data collection and collected data from 15,000 devices. Among them, there are 5,000 network cameras, 5,000 routers, and 5,000 printers, including 238 device brands, 231 network camera models, 161 routers models, and 251 printer models. In our experiments, the ratio of training data to test data is 9:1.

There are great differences in the information transmitted by different types of devices in the process of network communication. As we mentioned in the previous sections, the accuracy rate of Shodan and ZoomEye search engine for device type identification based on banner can achieve more than 95%. Therefore, our method is more focused on the identification of devices brand and models. We utilize the
device data collected to construct two data sets with different granularity labels. The first set of label granularity is the device brand, the second is the device model. We utilized these two data sets to implement model training through experiments using a variety of machine learning methods and to evaluate the device fingerprints we constructed, and compare them with current devices identification methods to verify the identification ability of our method.

A. DEVICES FINGERPRINT EVALUATION

We use three different neural network models to compare with our CNN+LSTM model, LSTM-only classifiers, CNN-only classifiers, and CNN+SVM classifiers. Through the comparison of multiple experiments, the parameters of our method are set as follows: The neural network model proposed in this paper is implemented by four convolutional layers, three pooling layers, two fully connected layers and LSTM network. Every size of kernels is set to $3 \times 3$ for convolution operation, the stride is set to 1, each size of pooling layer window is set to $2 \times 2$ and the max-pooling layer is selected. For the LSTM-only method, the LSTM+SoftMax structure is used to implement devices identification. In the CNN-only method, we utilize the CNN+SoftMax structure to implement devices identification, the CNN model includes three layers of convolutional layers and two layers of max-pooling layers. The first, second and third layers of convolution are implemented by 256 convolution kernels with the size of $3 \times 3$. The max-pooling layer window size is set to $2 \times 2$, and 50% dropout is chosen to avoid overfitting. The last layer is a fully connected layer. In the CNN+SVM model, the CNN structure is the same as the CNN-only method. Finally, we calculate the identification accuracy and recall rate of each method respectively, and use them as the criterion for judging the identification result. The calculation formula is as follows:

\[
\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \quad (10)
\]

\[
\text{recall} = \frac{TP}{TP + FN} \times 100\% \quad (11)
\]

where $TP$ is the number of true positives, $FP$ is the number of false positives, $FN$ is the number of false negatives, and $TN$ is the number of true negatives.

As shown in Figure 4, the identification performance of the CNN+LSTM neural network model is the better than others. In the final identification results, all these methods can reach the accuracy of more than 90%. Among them, the identification accuracy of CNN+SVM and CNN+LSTM models for device brand identification is more than 95%, and they can reach a higher $TP$ rate under the condition of low $FP$ rate. The performance of CNN+SVM model is better than CNN-only and LSTM-only methods. However, when the identification granularity is device model, the identification accuracy of CNN+SVM model decreases rapidly, even lower than that of CNN-only method.

In order to further demonstrate the identification performance of our proposed method for different device granularities, we randomly selected multiple groups of identification results from the first experiment, and used them to compare the identification performance of these neural network models for different device granularities. In Figure 5 (a), (b), we compare the identification performance of the four methods for device brand and model. The $X$-axis represents the accuracy of the identification results, and the $Y$-axis represents the recall rate of the identification results.

Each point in Figure (a) represents the identification accuracy and recall rate of the method to different brands of device. Each point in Figure (b) represent the identification accuracy and recall rate of each method to different device models. It can be shown from Figure (a), (b) that with the improvement of identification granularity, the identification accuracy of LSTM-only and CNN-only decreased. For different model of devices with the same type and brand, the device behavior is more similar and more difficult to identify. This is the difficulty that all the device identification methods need to overcome. The CNN+SVM and CNN+LSTM methods in figure 5 have an accuracy and recall rate of more than 90%, but the CNN+LSTM has better identification performance for device model. Because the former utilizes SVM
method, the training time of SVM model is more than that of CNN+LSTM model.

B. IDENTIFICATION PERFORMANCE EVALUATION
The time consumption of the device identification process is also an important indicator to evaluate the performance of device identification method. In order to evaluate the performance of our method, we identified the models of 15,000 devices and counted the identification results and time consumption.

In this paper, we compared the devices identification performance with the current UPnP-based [20], Banner-based [17] method and Nmap. Figure 6 shows the comparison of the accuracy, recall rate and time-consuming of each method. The X-axis is all the identification methods, the blue histogram and red histogram indicates the identification accuracy and recall rate, which correspond to the Y-axis on the left, and the yellow histogram indicates the time consumption of the method, which corresponds to the Y-axis on the right. The time consumption of our method is similar with that of the UPnP-based method and the banner-based method, although the accuracy difference among UPnP-based, Banner-based and HTTP+TCP method is small, as mentioned in the previous article, the application rate of UPnP protocol in IoT devices is lower than that of HTTP protocol. Since the banner information is often lost in the packets returned by the device, which leads to a low recall rate of Banner-based method. Our method can identify more classes of device and maintain an accuracy rate of more than 95%. It is more suitable for fine-grained device identification in large-scale networks. Moreover, compared with Nmap, the identification accuracy of the method we proposed is higher, with an average accuracy of 96.6% and an average recall rate of 97.4%, and the time consumption of our method is much less than Nmap, and similar with the UPnP-based and Banner-based methods.

VII. CONCLUSION
Nowadays, a large number of IoT devices are connected to the Internet, and IoT devices identification technology plays an important role in devices management and security. In this paper, we propose a proactive devices identification method that combines the neural network model to extract and analyze HTTP and TCP packets features and realize the fine-grain identification of IoT devices based on cross-layer protocol fingerprints. We have verified that our method has better identification accuracy and time-efficiency for large-scale fine-grained device identification. In the process of identifying device models, the average identification accuracy and recall rates of our method reached 96.6% and 97.4% respectively. Meanwhile, our time cost is limited in a reasonable range. But the disadvantage is that our proactive identification method is more dependent on the network state, which will rise the device identification time consumption.

REFERENCES
[1] *IoT Needs Next-Generation Infrastructure and the UK Aims to Provide It.* Accessed: Aug. 6, 2020. [Online]. Available: https://readwrite.com/2019/09/30/iot-needs-next-generation-infrastructure-and-the-uk-aims-to-provide-it/
[2] Z. Wang, J. Hu, Q. Wang, R. Lv, J. Wei, H. Chen, and X. Niu, “Task-Bundling-Based incentive for location-dependent mobile crowdsourcing,” *IEEE Commun. Mag.*, vol. 57, no. 2, pp. 54–59, Feb. 2019.
[3] M. Song, Z. Wang, Z. Zhang, Y. Song, Q. Wang, J. Ren, and H. Qi, “Analyzing user-level privacy attack against federated learning,” *IEEE J. Sel. Areas Commun.*, vol. 38, no. 10, pp. 2430–2444, Oct. 2020.
[4] A. Aksoy and M. H. Gunes, “Operating system classification performance of TCP/IP protocol headers,” in *Proc. IEEE 41st Conf. Local Comput. Netw. Workshops (LCN Workshops)*, Nov. 2016, pp. 112–120.
[5] Q. Li, X. Feng, L. Zhao, and L. Sun, “A framework for searching Internet-wide devices,” IEEE Netw., vol. 31, no. 6, pp. 101–107, Nov. 2017.

[6] J. C. Matherly. (2009). Shodan Search Engine. [Online]. Available: https://www.shodan.io

[7] H. J. Patel, M. A. Temple, and R. O. Baldwin, “Improving ZigBee device network authentication using ensemble decision tree classifiers with radio frequency distinct native attribute fingerprinting,” IEEE Trans. Rel., vol. 64, no. 1, pp. 221–233, Mar. 2015.

[8] T. Kohno, A. Broido, and K. Claffy, “Remote physical device fingerprinting,” in Proc. IEEE Symp. Secur. Privacy (S&P), May 2005, pp. 211–225.

[9] Y. Meidan, M. Bohadana, and A. Shabtai, “ProfiIoT: A machine learning approach for IoT device identification based on network traffic analysis,” in Association for Computing Machinery, New York, NY, USA, 2017, pp. 506–509.

[10] J. Caballero, S. Venkataraman, P. Poosankam, M. G. Kang, D. Song, and A. Blum, “FiFG: Automatic fingerprint generation,” in Proc. 14th Annu. Netw. Distrib. Syst. Secur. Symp. (NDSS), 2007, pp. 1–16.

[11] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, “Wireless device identification with radiometric signatures,” in Proc. 14th ACM Int. Conf. Mobile Comput. Netw. MobiCom, 2008, pp. 116–127.

[12] S. V. Radhakrishnan, A. S. Uluagac, and R. Beyah, “GTID: A technique for physical device and device type fingerprinting,” in IEEE Trans. Depend. Sec. Comput., vol. 12, no. 5, pp. 519–532, Oct. 2015.

[13] K. Gao, C. Corbett, and R. Beyah, “A passive approach to wireless device fingerprinting,” in Proc. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN), Jun. 2010, pp. 383–392.

[14] 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 Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS), Jun. 2017, pp. 2177–2184.

[15] J. A. Pinheiro, M. A. P. J. de BezerraCaio Burgardt, and R. D. Campelo, “Identifying IoT devices and events based on packet length from encrypted,” Comput. Commun., vol. 144, pp. 8–17, Aug. 2019.

[16] S. Bratus, C. Cornelius, D. Kotz, and D. Peebles, “Active behavioral fingerprinting of wireless devices,” in Proc. 1st ACMConf. WiSec, 2008, pp. 56–61.

[17] K. Yang, Q. Li, and L. Sun, “Towards automatic fingerprinting of IoT devices in the cyberspace,” Comput. Netw., vol. 148, pp. 318–327, Jan. 2019.

[18] A. R. Khakpour and A. X. Liu, “An information-theoretical approach to high-speed flow nature identification,” IEEE/ACM Trans. Netw., vol. 21, no. 4, pp. 1076–1089, Aug. 2013.

[19] Zoomeye. Accessed: Feb. 6, 2020. [Online]. Available: https://zoomeye.org/

[20] UPnP. Accessed: Feb. 6, 2020. [Online]. Available: https://openconnectivity.org/developer/specifications/UPnP-resources/UPnPArchitectura

[21] B. Bezawada, M. Bachani, J. Peterson, H. Shirazi, I. Ray, and I. Ray, “IoTSense: Behavioral fingerprinting of IoT devices,” in Proc. Workshop, 2018, pp. 41–50.

[22] Censys. Accessed: Feb. 6, 2020. [Online]. Available: https://censys.io/

DAN YU was born in Taiyuan, China, in 1983. She received the B.S. degree in electronics engineering from the North University of China in 2007 and the M.S. degree in electronics engineering from the Beijing University of Posts and Telecommunications in 2013. She is currently pursuing the Ph.D. degree with the College of Information and Computer, Taiyuan University of Technology, Taiyuan.

Her research interests include wireless sensor networks and the Internet of Thing (IoT) security.

HAOGUANG XIN was born in Datong, China, in 1996. He received the B.S. degree in software engineering from the Taiyuan University of Technology, Taiyuan, China, in 2018, where he is currently pursuing the master’s degree with the College of Information and Computer. His current research interests include device identification and the IoT security.

YONGLE CHEN (Member, IEEE) was born in Weifang, China, in 1983. He received the B.S. degree from Jilin University in 2007, the M.S. degree from the Institute of Software, Chinese Academy of Sciences, in 2009, and the Ph.D. degree in computer science from the University of Chinese Academy of Sciences in 2013, all in computer science.

From 2013 to 2015, he was an Assistant Professor with the College of Information and Computer, Taiyuan University of Technology, Taiyuan, China, where he has been an Associate Professor since 2015. His research interests include wireless sensor networks, indoor positioning, and the IoT security.

YAO MA was born in Taiyuan, China, in 1982. He received the B.S. and M.S. degrees in computer science from the Beijing Institute of Machinery Industry, China, in 2004 and 2007, respectively, and the Ph.D. degree in information technology from Towson University, USA, in 2012.

Since 2012, he has been an Assistant Professor with the College of Information and Computer, Taiyuan University of Technology, Taiyuan. His research interests include universal usability, indoor positioning, and Web security.

JUNJIE CHEN was born in Taiyuan, China, in 1956. He received the B.S. and M.S. degrees in computer science from Shanghai Jiao Tong University and the Ph.D. degree in computer applications from the Beijing Institute of Technology.

He is currently a Full Professor with the College of Information and Computer, Taiyuan University of Technology, Taiyuan, China. His research interests include intelligent computing and the IoT security.

***