Accurate Identification of IoT Devices in the Presence of Wireless Channel Dynamics

Bhagyashri Tushir, Vikram K Ramanna, Yuhong Liu, Behnam Dezfouli
Internet of Things Research Lab, Department of Computer Science and Engineering, Santa Clara University, USA
{btushir, vramanna, yhliu, bdezfouli}@scu.edu

***Abstract*—Identifying IoT devices is crucial for network monitoring, security enforcement, and inventory tracking. However, most existing identification methods rely on deep packet inspection, which raises privacy concerns and adds computational complexity. Moreover, existing works overlook the impact of wireless channel dynamics on the accuracy of layer-2 features, thereby limiting their effectiveness in real-world scenarios. In this work, we define and use the latency of specific probe-response packet exchanges, referred to as "device latency," as the main feature for device identification. Additionally, we reveal the critical impact of wireless channel dynamics on the accuracy of device identification based on device latency features. Specifically, this work introduces "accumulation score" as a novel approach to capturing fine-grained channel dynamics and their impact on device latency when training machine learning models. We implement the proposed methods and measure the accuracy and overhead of device identification in real-world scenarios. The results confirm that by incorporating the accumulation score for balanced data collection and training machine learning algorithms, we achieve an F1 score of over 97% for device identification, even amidst wireless channel dynamics, a significant improvement over the 75% F1 score achieved by disregarding the impact of channel dynamics on data collection and device latency.**

***Index Terms*—Security, Wi-Fi, Channel Utilization, Machine Learning, Privacy, Feature Collection**

I. Introduction

The rapid growth in the number and variety of Internet of Things (IoT) devices underscores the critical need for precise device identification, which empowers network middleboxes and appliances (such as wireless Access Points (APs), switches, and network controllers) to enhance the management and security of connected devices [1], [2]. For example, micro-segmentation strategies can be employed to segregate devices based on their functions and security requirements [3]. Additionally, should a device exhibit unusual traffic pattern indicative of potential security threats [4], it can be temporarily isolated for investigation. Moreover, accurate identification of IoT devices enables fine-tuning connectivity settings tailored to the specific needs of each device [5]. This customization prioritizes devices with higher importance or specific latency demands and significantly improves the overall user experience. Beyond operational efficiency, IoT device identification offers valuable insights into device usage, revealing patterns and behaviors instrumental in enhancing existing services or inspiring new product developments. Proactively monitoring devices' performance and status through identification also aids in the early detection of potential malfunctions, allowing for timely interventions to prevent failures.

The conventional techniques for identifying IoT devices, primarily based on IP and MAC addresses, are inadequate due to their limited applicability and susceptibility to security threats such as spoofing [6]. Various recent works have leveraged the unique traffic patterns of IoT devices [7], [1], [8], [9], [10], [11], [12], [13], [14]. However, these studies have some fundamental limitations. Firstly, their proposed features rely on specific network configurations and traffic conditions, limiting their applicability over time and across different deployment environments. For instance, while features such as packet length and packet rate are impacted by the dynamics of traffic patterns, wireless channel, and the number of devices accessing the wireless channel, such impacts have been overlooked [7], [11]. Secondly, the process of extracting features at various levels of the protocol stack (including attributes like destination DNS queries, IP addresses, and port numbers) not only risks compromising user privacy due to the sensitivity of the information involved but also results in increased computational complexity and memory demands [7]. Thirdly, existing approaches involve collecting training data over one or several weeks, causing a long delay before the data can be used for device identifications [7].

In light of the identified challenges, this paper is guided by two key design considerations: (i) given the sensitivity of IoT devices (such as smart homes where personal data is handled within the privacy of individual homes), it is imperative to identify these devices in a manner that safeguards users' personal information; (ii) the dynamic nature of deployment environments, characterized by evolving traffic patterns of devices and fluctuating wireless channel properties. Addressing these critical concerns, this paper makes the following contributions towards more reliable identification of Wi-Fi (802.11) IoT devices. We introduce measurement techniques and features based on device response times to TCP and UDP probe packets, enabling the fingerprinting of various IoT device types. While latency features are useful for privacy-preserving device fingerprinting, our analysis reveals that such features are significantly influenced by wireless channel dynamics, limiting their stability and accuracy under various wireless channel conditions. Although Channel Utilization (CU) is a common metric for assessing wireless channel dynamics, its granularity and reliability are hampered by inherent short-term dynamicity and hardware limitations. To overcome these limitations, we introduce *accumulation score* as a robust metric for instantaneous measurement of channel dynamics and their influence on device latency. This metric provides a more reliable approach for understanding and adapting to changing network conditions. We implement the proposed methods and build a testbed to measure identification accuracy and system's overhead in real-world settings. Our empirical evaluations demonstrate that device identification methodologies that fail to account for channel dynamics yield accuracy rates ranging from 25% to 75%. In contrast, by incorporating the accumulation score into our data collection and Machine Learning (ML) algorithm training processes, we achieve identification accuracy exceeding 97% in the presence of diverse wireless channel conditions. We will also present training and inference overhead analysis of the ML algorithms when used on a residential AP.

The rest of the paper is organized as follows. In Section II, we study the opportunities and challenges of using device latency for device identification. In Section III, we present accumulation score and its importance for device identification. We present empirical performance evaluation of device identification accuracy and overhead of ML algorithms in Section IV. We overview related work in Section V and conclude the paper in Section VI.

II. Significance of Device Latency

In this section, we present the motivation behind considering *device latency* as a feature for IoT device (hereinafter referred to as *device*) identification. We also identify the challenges of measuring this feature and discuss our proposed methodology. Finally, we illustrate how device latency varies among devices across different CU ranges.

A. Motivation for Utilizing Device Latency

In this work, we adopt *device latency* (denoted as l) as the primary feature for device identification in Wi-Fi networks. Device latency is defined as follows: *once a device receives a packet, how long it takes to process the packet, generate a response, and start the transmission of the response packet.* This paper refers to the packets used to measure device latency as *probe packets*. Referring to Figure 1, t_4 denotes the time instance at which the device has wholly received a probe packet, and device latency is the interval between t_4 to t_6. Specifically, device latency comprises the following two components: response generation and channel contention, corresponding to time intervals t_4 to t_5 and t_5 to t_6, respectively.

Several factors contribute to the variability of these latency components, including the packet type, the device's hardware characteristics such as processor type and memory, the type of wireless Network Interface Card (NIC) employed, NIC's driver, operating system and the network stack implemented on the device. These attributes collectively influence how a device handles latency-related tasks. For instance, the processor, NIC driver, operating system, and network stack impact incoming and outgoing packet processing, while the wireless NIC and its driver determine how a device contends for channel access before transmitting each packet.

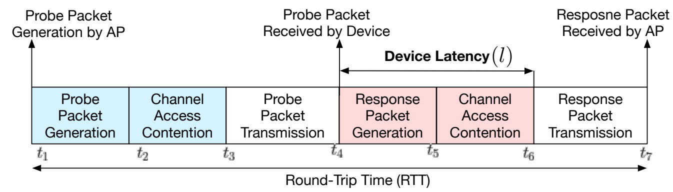

Fig. 1. Device latency (l) is defined as the interval between t_4 to t_6. Round-Trip Time (RTT) is defined as the interval between crafting a probe packet by the AP until the reception of the response packet (generated by the device) by the AP. Extracting device latency (t_4 to t_6) from RTT (t_1 to t_7) is challenging due to the variability of t_1 to t_4.

B. Feature Measurement Methodology

The precise measurement of device latency is essential for accurate device identification. One potential method is using Round-Trip Time (RTT), which includes the various time intervals illustrated in Figure 1: (i) packet generation interval, from t_1 to t_2, which represents the time taken by the AP to craft a probe packet, (ii) channel access contention by the AP during t_2 to t_3 to send the probe packet, (iii) actual transmission of probe packet from t_3 to t_4, (iv) packet processing and response generation during t_4 to t_5 by the device, (v) channel access contention by the device during t_5 to t_6 to send the response packet, and (iv) actual transmission of response packet during t_6 to t_7. Among these delay components, the time interval from t_1 to t_4 does not represent device latency. It can be easily impacted by different factors, such as the packet switching load of the AP or the network traffic. Therefore, the variability of this time interval can negatively affect device latency measurement accuracy, impacting the effectiveness of device identification.

We propose an alternative method to address the above challenges. Instead of relying on the RTT method to measure the time intervals encompassing t_1 to t_7, we focus on the interval from t_4 to t_7 and subtract t_6 to t_7 to extract device latency. To this end, we employ a packet sniffer to capture probe packets and the corresponding response packets. A program then parses the captured data and correlates probe and response packets for each device. Detailed packet information, such as packet type and duration, can be extracted from the captured data to compute the proposed features. For instance, if the probe packet is a TCP-SYN sent to a closed port, the packet parser matches the TCP-SYN with a TCP-RST packet by correlating their sequence number fields. Once a pair of packets (probe and response) is found, the packet parser extracts t_4 as the time of receiving the probe packet and t_7 as the time of receiving the response packet. Then, the duration of the response packet ($t_7 - t_6$) is subtracted from $t_7 - t_4$ to extract device latency.

It is worth noting that the packet capture process does not necessarily require a sniffer in addition to the AP. For example, many modern APs provide an additional NIC that can be used for capturing packets on the desired channel. From the

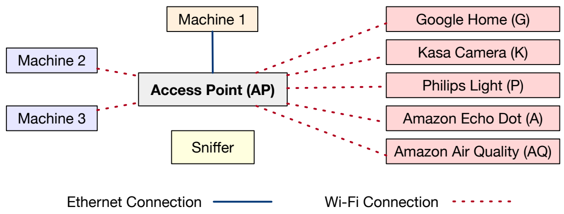

Fig. 2. The testbed components including machines used for background traffic generation, IoT devices, and sniffer. Note that various experiments of this work utilize subsets of these components, depending on the specific objectives and requirements of each study.

memory utilization perspective, considering that Wi-Fi APs typically have limited storage capacity, the proposed approach is designed to optimize storage use. For example, the Netgear WAX218 has 256 MB of flash memory and 512 MB of RAM. The parser program processes captured packet headers in real-time and selectively stores only essential information from the packets of interest. It is also essential to note that the packet parser only stores packet latency information and probe packet type for each device. Therefore, no user-sensitive information (such as destination IP addresses) is stored and used by the identification algorithm. This approach ensures that the proposed method is privacy-preserving.

C. Device Latency versus Device Type

We empirically validate the significance of device latency in two essential aspects: its role in facilitating device identification and its sensitivity to variations in CU intensity. To this end, we conduct preliminary experiments within a smart home testbed, as shown in Figure 2. The testbed includes off-the-shelf devices, each denoted by the notation enclosed in parentheses: Google home (G), Amazon echo dot (A), Kasa camera (K), Philips light bulb (P), and Amazon air quality monitor (AQ). All of these devices are connected to the AP. Machine 1 is connected to the AP's Ethernet port, and Machine 2 and Machine 3 are connected to the AP wirelessly. The AP monitors CU intensity using the `ethtool` utility, which captures measurements at 10 millisecond (ms) intervals (shortest possible interval) to determine the percentage of time the channel remains occupied during each interval [15]. To evaluate the impact of CU on device latency, we adjust CU intensity (from 0% to 100%) by using `iperf` and varying the data rates of the flows being exchanged between Machine 1 and Machine 2 and Machine 3 via. To elicit responses from the devices, the AP transmits one low-payload (0 byte) TCP-SYN probe packet per second to a closed port of each device, anticipating a corresponding TCP-RST response packet.

Figures 3 (a) through (e) present statistical metrics encompassing the median, minimum, and maximum device latency for various devices and CU ranges. Device latency is denoted as l_{tcp}^{lo} because of the use of TCP-SYN probe packets with 0-byte (low) payload size. The statistical metrics are shown for different ranges of CU intensity, corresponding to the five sub-figures. Three primary observations can be made from Figure 3. Firstly, within each CU intensity range, the median device latency varies for each device. For instance, in Figure 3(a), the air quality monitor (AQ) shows the highest median device latency at 3.27 ms, while Google home (G) exhibits the shortest median latency at 0.62 ms. Secondly, significant differences exist in the minimum and maximum device latency values across all devices, with some overlapping within these metrics. These differences stem from variations in device hardware and software characteristics. Thirdly, when comparing the figures, we observe a simultaneous shift in median, minimum, and maximum device latency with changes in CU intensity across all devices. This relationship between CU intensity and device latency highlights the significance of considering both factors when addressing device identification.

Based on these findings, it becomes evident that leveraging device latency as the key feature for device identification has a significant potential. However, relying solely on 0-byte TCP-SYN probe packets may be insufficient for accurate device identification. With these insights, we introduce four distinct probe packet types to collect device latency features: low-payload (0 B) TCP-SYN, high-payload (1400 B) TCP-SYN, low-payload (0 B) UDP, and high-payload (1400 B) UDP. The respective device latency associated with these packet types are labeled as l_{tcp}^{lo}, l_{tcp}^{h}, l_{udp}^{lo}, and l_{udp}^{h}. The motivation for utilizing different payload sizes lies in the fact that larger payloads have an evident impact on various essential stages within the packet processing workflow, such as memory allocation and deallocation for packets, and Direct Memory Access (DMA) of packets from the NIC to the driver's receiver buffer. Therefore, exploring how device latency varies across these packet types and sizes yields insights into the latency variation among devices, enabling us to differentiate devices effectively. Note that the AP sends all these probe packets to the closed ports of the devices. Responses to UDP and TCP-SYN probes are ICMP destination unreachable packets and TCP-RST packets, respectively.

III. Accounting for the Impact of Channel Utilization on Device Latency

Some wireless NICs can measure and report CU by dividing channel activity time by a reference time period. For instance, Atheros NICs can measure CU every 10 ms [15], [16]. Since device latency can often be under 10 ms, as shown in Figure 3, and given the dynamic nature of CU, the coarse granularity of the CU measurements performed by these drivers is insufficient to accurately quantify the impact of CU on device latency. This limitation is increasingly significant in light of the higher physical layer rates introduced by new standards like 802.11be and the use of faster processors on IoT devices. Therefore, we propose a novel approach for a more granular and accurate analysis of the intensity of CU and its impact on each probe-response exchange.

A. Accumulation Score

In this section, we present the formulation of the *accumulation score*, a novel metric to measure instantaneous CU and its impact on device latency. We use the notation p_i

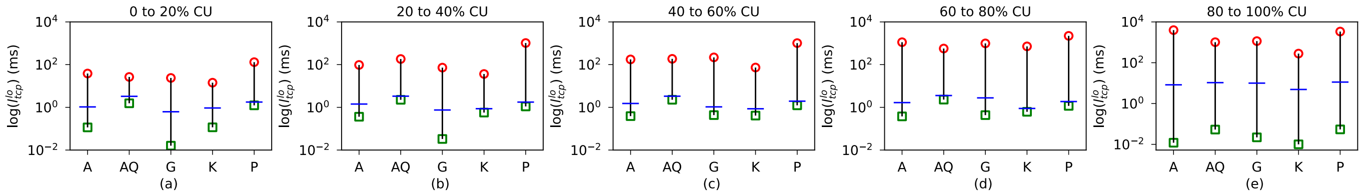

Fig. 3. Device latency (l_{tcp}^{lo}) for various devices and CU ranges. Results are collected when using low-payload (0 B) TCP-SYN probe packets. Circles, squares, and horizontal lines represent maximum, minimum, and median values, respectively.

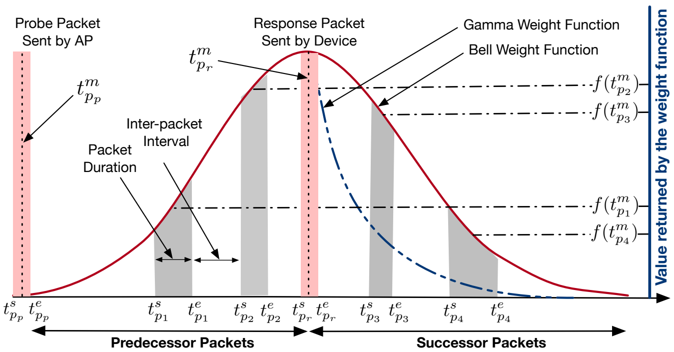

Fig. 4. A sample probe-response packet exchange with Predecessor and Successor packets. Accumulation score calculation associated with each device latency value uses three major parameters: the duration of Predecessor and Successor packets, inter-packet intervals, and value obtained from the projection of packets' midpoints onto the Bell or Gamma function.

to denote a generic packet. Specifically, probe and response packets are represented as p_p and p_r, respectively. For each packet p_i, we denote its start and end times as $t_{p_i}^{s}$ and $t_{p_i}^{e}$. We begin by introducing two essential concepts employed in the accumulation score formulation: (i) *Predecessor Packets* and (ii) *Successor Packets*. Figure 4 illustrates these concepts in the context of a probe-response exchange. *Predecessor Packets* are defined as the packets sent during the time interval between the transmission of the probe packet and the transmission of the response packet. In Figure 4, packets p_1 and p_2 are examples of Predecessor Packets. The occurrence of these packets indicates competition for channel access, potentially increasing the CU and resulting in increased device latency, particularly when these packets are transmitted close to the transmission of response packet. On the other hand, *Successor Packets* refer to the packets transmitted after the response packet, up until a specific time instance that will be further detailed later in this section. Packets p_3 and p_4 in Figure 4 are examples of Successor Packets. The rationale for considering Successor Packets lies in their potential impact on device latency, especially when a series of packets immediately follows the response packet. A cluster of such packets can suggest possible contention at the time the device was trying to transmit the response packet, thereby influencing device latency.

In evaluating the impact of CU on device latency, we focus on certain characteristics of Predecessor and Successor packets, including packet duration, mid-packet points, and inter-packet intervals. The duration of a packet, indicating the length of time it occupies the communication channel, is defined as $d_{p_i} = t_{p_i}^{e} - t_{p_i}^{s}$. Longer packet durations lead to extended channel occupancy and longer device latency. The midpoint of a packet's transmission, defined as $t_{p_i}^{m} = t_{p_i}^{s} + \frac{t_{p_i}^{e} - t_{p_i}^{s}}{2}$, represents the time at the center of the packet transmission. The inter-packet interval is defined as $v_{p_{i-1},p_i} = t_{p_i}^{s} - t_{p_{i-1}}^{e}$ and refers to the interval between packets p_{i-1} and p_i. Shorter inter-packet intervals typically signify higher CU, increased contention for channel access, and longer device latency.

To account for the impact of each Predecessor or Successor packet on CU, we consider the relative distance (time difference) of each packet from the response packet (t_{p_r}). The closer a packet is to its corresponding response packet, the higher its impact on device latency. To reflect this relationship, we define and use *weight functions* that reach their peak at the time of response packet transmission ($t_{p_r}^{m}$). Figure 4 demonstrates two such weight functions, called the Bell and Gamma curves. The Bell curve has a mathematical form similar to the Probability Density Function (PDF) of the normal distribution and is expressed as $f(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$, where x is the variable of interest, μ is the mean, and σ is the standard deviation. Similarly, the Gamma curve is expressed as $f(x;\alpha,\beta) = \frac{1}{\beta^{\alpha}} x^{\alpha-1} e^{-\frac{x}{\beta}}$, where α and β are parameters that control the shape and scale of the function. Note that the mean of both functions is at $t_{p_r}^{m}$, the middle point of the response packet. A gamma curve considers both distance and precedence, meaning that for a Predecessor and a Successor packet both equidistant from $t_{p_r}^{m}$, a higher weight is given to the Predecessor packet. In contrast, the Bell curve takes into account distance only when assigning weight. It is important to note that these weight functions assign relative weights to each packet, reflecting their contribution to CU intensity within a probe-response exchange rather than modeling the actual packet distribution.

To integrate the above metrics and evaluate the impact of Predecessor and Successor packets on each probe-response exchange, we introduce the *accumulation score* metric. Assume the number of Predecessor and Successor packets, denoted as n, may vary for each probe-response exchange. Accumulation score is defined as follows:

$$\mathcal{A}_j = \begin{cases} 0 & \text{if } n = 0 \\ d_{p_1} \times \sigma \times f(t_{p_1}^{m}) & \text{if } n = 1 \\ d_{p_1} \times \sigma \times f(t_{p_1}^{m}) + & \text{if } n > 1 \\ \quad \sum_{i=2}^{n} (d_{p_i} \times \frac{1}{v_{p_{i-1},p_i}/\sigma} \times f(t_{p_i}^{m})) & \end{cases} \quad (1)$$

TABLE I
PROPOSED FEATURES FOR DEVICE IDENTIFICATION

Symbol	Definition
l_{tcp}^{lo}	Device latency for low-payload (0 B) TCP-SYN probe
$\mathcal{A}_{tcp}^{lo}$	Accumulation score for low-payload (0 B) TCP-SYN probe
l_{tcp}^{h}	Device latency for high-payload (1400 B) TCP-SYN probe
$\mathcal{A}_{tcp}^{h}$	Accumulation score for high-payload (1400 B) TCP-SYN probe
l_{udp}^{lo}	Device latency for low-payload (0 B) UDP probe
$\mathcal{A}_{udp}^{lo}$	Accumulation score for low-payload (0 B) UDP probe
l_{udp}^{h}	Device latency for high-payload (1400 B) UDP probe
$\mathcal{A}_{udp}^{h}$	Accumulation score for high-payload (1400 B) UDP probe

Here, $f(\cdot)$ refers to the value returned by the weight function. If $n = 0$, we assign a zero value to the accumulation score. For $n = 1$, since there is no inter-packet interval, we normalize the duration of the packet by multiplying it by the standard deviation of the curve, denoted as σ. When selecting a weight function, we select function parameters that maximize the correlation between accumulation score and device latency. For $n > 1$, we further consider the impact of the inter-packet interval on the accumulation score for all subsequent packets beyond the first, ensuring that the influence of each packet on the score intensifies as their inter-packet intervals decrease. When normalizing the duration of a packet, we calculate the ratio of the inter-packet interval between the current packet and the subsequent one to the standard deviation (σ) of the weight curve. This is represented as $v_{p_{i-1},p_i}/\sigma$.

B. Accumulation Score for Various Probe Packet Types

As discussed earlier, various probe packets trigger different packet processing paths of devices. For instance, a TCP-SYN packet requires more processing compared to a UDP packet. Also, various packet sizes induce different overheads in terms of reception processing. For instance, the reception of a larger packet requires a longer reception time due to longer NIC to memory transfer and buffer management. Based on these observations, we use device latency and accumulation scores for the four probe packets explained in Section II-C. Table I summarizes the device latency and accumulation score features we use to train the ML algorithms.

Following data collection, we determine the parameters of the Bell and Gamma functions for each of the aforementioned packet types to maximize the Pearson correlation between device latency and accumulation score. We found correlation values of 92% and 87% for the Bell and Gamma weight functions, respectively. Consequently, we employ the Bell curve to determine the accumulation score considering its higher correlation.

C. Effectiveness of Representing CU via Accumulation Scores

To validate the effectiveness of the accumulation score as a tool for modeling CU, we conduct an analysis estimating the accumulation score of various probe packets across different CU ranges. The details of data collection for this analysis can be found in Section IV. The outcomes of this analysis are depicted in Figures 5 (a) to (d). A key observation from Figure 5 is the trend of increasing accumulation score values directly related to CU levels. This trend is consistent across all four types of probe packets. Specifically, we note that lower CU ranges, particularly from 0 to 20%, are characterized by lower accumulation scores, while the highest CU range (80 to 100%) corresponds with the maximum accumulation scores. This pattern indicates a positive correlation between the accumulation score and CU, suggesting that higher CU is associated with higher accumulation scores. Also, our regression analysis demonstrates that an exponential function captures the relationship between the accumulation score and CU. This is evidenced by a Residual Sum of Squares (RSS) value of less than 0.004, indicating a strong fit.

Although a consistent increase in accumulation scores can be observed with the increase of CU for all four probe packet types, there are overlapping values of accumulation scores for different CU ranges. This observation is due to the highly-dynamic nature of CU and accumulation score, which result in overlaps of collected data instances. While we strive to control the CU values by generating background traffic (exchanged between Machines 1, 2, and 3), such control is still very rough and cannot ensure that the instantaneous CU values are always accurately or stably enforced.

Figure 6 shows the distribution of device latency for various ranges of accumulation score values and all the IoT devices in our testbed. We adopt three bins of varying lengths to account for the fluctuations observed in accumulation scores and the presence of a long tail in the accumulation score distribution. It is evident from Figure 6 that an increase in the accumulation score corresponds to a rise in device latency. This further validates the effectiveness of using accumulation scores to represent CU.

IV. EMPIRICAL EVALUATIONS AND DISCUSSION

In this section, we demonstrate the importance of using accumulation scores in data collection and model training. Also, we study the impact of using various subsets of features on accuracy and reveal the effectiveness of the proposed features and various ML algorithms in identifying devices.

A. Impact of Accumulation Score and Channel Utilization

To demonstrate the influence of CU on device identification accuracy, we use Light Gradient Boosted Machine (LGBM) algorithm and train the model within predefined accumulation score ranges and subsequently assess the performance of device identification across different accumulation score ranges. Each defined accumulation score range bin comprises approximately 11,000 samples. We adopt a progressive training approach in our analysis, where we start the training process with an initial set of 1,000 samples and then gradually increase the training dataset size, adding samples incrementally until reaching 11,000. Throughout this process, we maintain a constant number of test samples at 1,000 for each iteration.

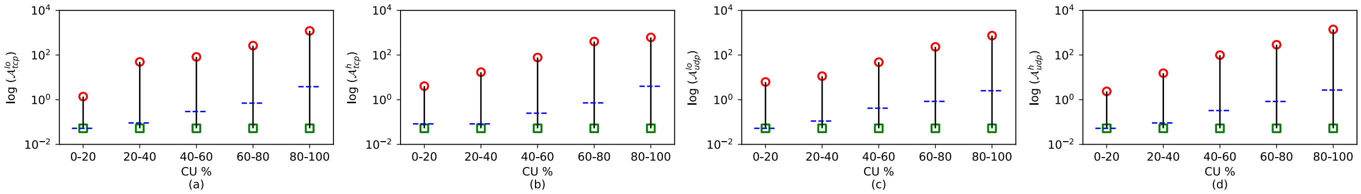

Fig. 5. Illustration of different accumulation scores for various CU ranges and probe packets. With reference to Table I, each sub-plot represents a specific accumulation score: (a) $\mathcal{A}^{\text{lo}}_{\text{tcp}}$, (b) $\mathcal{A}^{\text{h}}_{\text{tcp}}$, (c) $\mathcal{A}^{\text{lo}}_{\text{udp}}$, and (d) $\mathcal{A}^{\text{h}}_{\text{udp}}$. The y-axis is log-scaled to facilitate a better understanding of the relationship between accumulation scores and the ranges of CU. Circles, squares, and horizontal lines represent maximum, minimum, and median values, respectively. These results confirm the direct relationship between the accumulation score and CU.

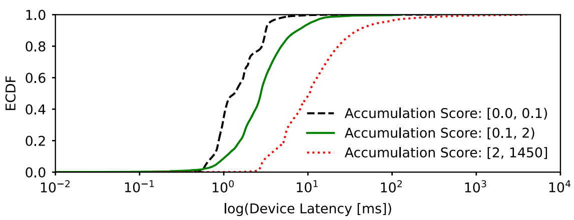

Fig. 6. Device latency (l) distribution across all devices and various accumulation score ranges. With the increase in accumulation score, the device latency also increases.

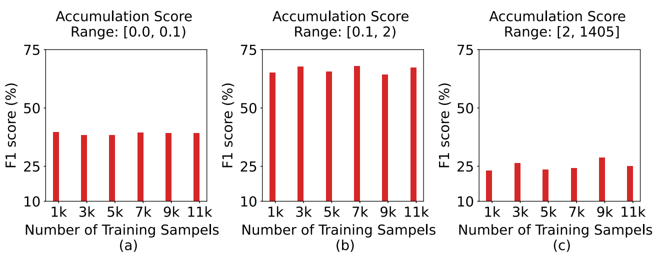

Fig. 7. F1 score of device identification accuracy when the LGBM algorithm is trained using the samples belonging to one accumulation score range but tested with samples belonging to the other two accumulation score ranges. The low F1 scores highlight the necessity of involving a diverse range of accumulation scores in the training data.

The division of the accumulation score range is structured into three distinct intervals: $[0, 0.1)$, $[0.1, 2)$, and $[2, 1405)$. These intervals are approximately mapped to the corresponding CU ranges of $[0, 33\%)$, $[33\%, 66\%)$, and $[66\%, 100\%)$. This mapping is established through the collection of accumulation score samples under varying CU intensities, and then employing a regression analysis to correlate the accumulation scores with CU. The CU range is then segmented into three intervals based on this correlation, with each interval's starting and ending points derived from the corresponding segments of the regression line.

Figures 7 (a) through (c) present the variation of the F1 score as a function of different training sample sizes, with the x-axis depicting the number of training samples used. The analysis reveals a notable trend: when the algorithm is trained exclusively within a specific accumulation score range, the F1 score consistently falls below 75%, sometimes even dropping to as low as 25%. This pattern persists despite increases in the size of the training dataset, with F1 scores remaining under the 75% threshold. This finding underscores an important limitation: *algorithms trained on data from a singular CU range exhibit poor generalization when applied to device identification tasks across different CU ranges.* The model effectively learns and recognizes patterns only within the narrow confines of its training range; consequently, when confronted with CU levels outside this familiar range, the model's ability to accurately identify devices diminishes. Merely increasing the training dataset within the same accumulation score range does not rectify this issue, as it does not expose the model to the broader spectrum of patterns associated with different CU levels. We also observe a relative improvement in accuracy when the algorithm is trained using the middle accumulation score interval (i.e., $[0.1, 2)$). We attribute this enhanced performance to the overlap of CU samples from lower and upper intervals within the middle range. As a result, the model gains partial exposure to device behaviors across the broader spectrum of CU ranges, and this exposure enables the algorithm to partially understand device patterns outside its primary training interval.

To further validate the significance of utilizing accumulation score for device identification, we assess the performance of LGBM algorithm across various feature sets and present the results in Figures 8. We utilize a randomized dataset partitioning method to evaluate the algorithm's robustness and mitigate the effects of potential data variability on performance. This procedure is repeated by increasing the number of samples. We iterate this process 30 times for each sample size. Compared to Figure 7, the results presented in Figures 8 represent higher accuracy when the model has been trained with samples spanning the complete range of accumulation scores (and CUs). Therefore, as expected, the F1 scores are considerably higher than those presented in Figure 7. For instance, Figure 7 (c) shows F1 scores of around 25% when all the latency and accumulation score features are used. Therefore, these results confirm the importance of model training across the range of accumulation scores.

The results in Figures 8 confirm that even when the model has been trained across the range of accumulation scores, including the accumulation score as a feature consistently yields improved F1 scores. Also, the accumulation score allows us to achieve higher F1 scores when some latency features are ignored or when the number of samples is limited. For instance, consider a scenario where only UDP features can be used instead of TCP. Such restrictions may be enforced for reasons such as UDP's lower overhead on devices' resource

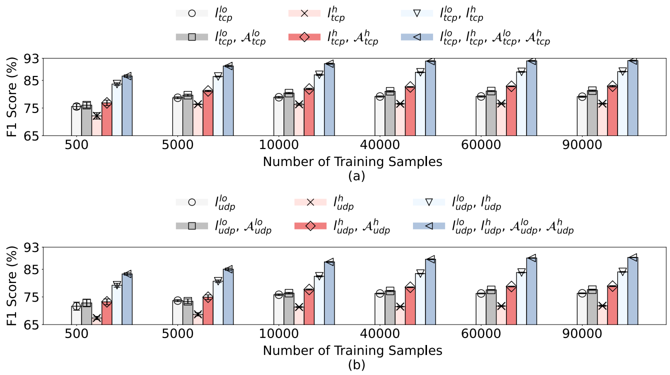

Fig. 8. The F1 scores achieved by the LGBM algorithm, with each score corresponding to the model trained on different subsets of features. The legend's columns represent various combinations of device latency features and their associated accumulation score features. Error bars represent standard deviation for 30 iterations.

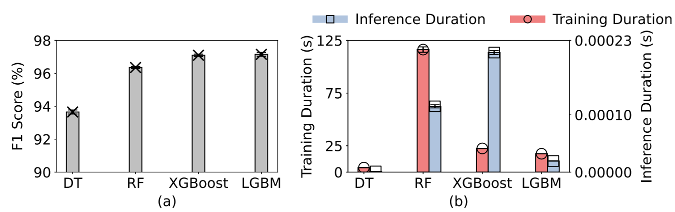

Fig. 9. (a): F1 score of various ML algorithms. (b): Duration of training (circles) and inference (square). Error bars represent standard deviation for 30 iterations.

consumption compared to TCP or the restrictions on sending non-legitimate, crafted TCP-SYN packets in a network due to security or firewall configurations. In this case, the extraction and use of associated accumulation scores features with UDP latency features result in considerable performance improvement. For instance, when we have 500 samples, Figure 8(b) shows that using feature vector $\mathbf{x} = [l^{lo}_{udp}, \mathcal{A}^{lo}_{udp}]$ instead of feature vector $\mathbf{x} = [\mathcal{A}^{lo}_{udp}]$ increases the F1 score from 67% to 73%. It is important to note that calculating the accumulation score for each latency feature is implicit and does not impose any additional data exchange overhead.

B. Performance Validation via Different ML Algorithms

Next, we further validate the effectiveness of the proposed features by comparing the accuracy and overhead of various ML algorithms, including Decision Tree (DT), Random Forest (RF), LGBM and XGBoost. We employ random dataset partitioning to understand the impact of training data variations on model performance and robustness. Specifically, we utilize the whole dataset, including 125,336 samples, and partition it into distinct training and testing sets with an 80:20 split ratio. This process is iterated 30 times.

Figure 9(a) shows the accuracy of various ML algorithms. DT demonstrates the lowest F1 score of 93.6%, whereas, LGBM and Extreme Gradient Boosting (XGBoost) achieve the highest score of around 97%. Figure 9(a) also shows the low standard deviation of F1 scores, indicating low bias and variances in these ML algorithms. It further suggests that the proposed features are robust and stable across different ML algorithms and not influenced by the dataset's random partitioning.

The Netgear WAX 218 AP includes an IPQ8074 SoC [17], which integrates a quad-core Cortex-A53 processor. Using this processor, we measure the duration of training and inference and present the results in Figure 9(b). The results show that DT is the fastest model to train and has the shortest inference duration, suggesting a suitable solution for resource-constrained scenarios. Considering the two models that achieve the highest F1 scores, the inference duration of LGBM is significantly shorter than that of XGBoost. Therefore, if offline training can be conducted, using LGBM is specifically beneficial for resource-constrained devices, such as home APs.

V. Related Work

Recent works on IoT device identification center around leveraging traffic-based features to achieve high accuracy. We have classified these studies into a few distinct categories, determined by the types of features they utilize.

The first category of features is extracted by analyzing individual network packets. For example, [18] utilized TLS protocol features, such as TLS version, server name, IP address, and negotiated cipher suites, for device identification. Similarly, the study in [19] adopted features like UDP checksum, TCP window size, DNS query class, IP length, TCP port, and TCP flags, focusing on packet headers and network traffic. Expanding the scope, [10] extracted features across 16 protocols, including HTTP, DNS, and NTP, analyzing packet size, port numbers, and IP address counts for each protocol type. Moreover, [8] focused on features such as packet length and the count of specific packet types, including STUN, NTP, and MQTT. While these studies may achieve high accuracy in device identification, their reliance on analyzing packet contents raises privacy and computational resource concerns. Also, stringent data protection regulations could further restrict access to such packet contents, complicating the analysis.

The second category of features emphasizes flow-based statistics aggregated across packets. For example, [12] proposed flow-based features like the percentages of ICMP, TCP, and UDP protocols, packet count and size, and the diversity ratio of IP addresses. Shaikh et al. [20] utilized metrics such as average time-to-live per IP address, total packet count, unique IP addresses and port counts, and TCP/UDP packet numbers. Cvitic et al. [21] focused on the length of both sent and received packets. Features like packet and byte counts, mean packet size, inter-arrival time, and the count of TCP packets with specific flags were used in [22]. This approach was refined in [1] and [11], where the former used packet size, inter-arrival time, and specific TCP flag counts, while the latter tailored their analysis to encrypted traffic, examining packet length statistics over a one-second window. In [23], the study identified IoT devices in Wi-Fi environments, emphasizing feature selection from the 802.11 link layer, including frame length, type, arrival time, and retransmission flag. Despite the strengths of flow-based features, gathering comprehensive training data to represent flow statistics accurately can be challenging and time-consuming, especially for IoT devices that operate intermittently and under resource constraints. Furthermore, these flow-based features can be sensitive to wireless channel conditions, significantly limiting their generality across different network environments. For instance, as demonstrated in this paper, high CU during peak times can alter flow patterns, impacting identification accuracy.

Several studies [9] have combined flow-based features with packet-specific features to leverage the benefits from both categories and improve identification accuracy further. For example, both individual packet features, such as domain names, cipher suites, DNS intervals, and port numbers, as well as flow-level features, like the total number of bytes exchanged, the duration of flows, and the inactive intervals of IoT devices, are examined in [7]. Although these studies lead to continuous improvement in identification accuracy, there is a lack of solutions that can address privacy concerns and features' reliance on network conditions at the same time.

VI. Conclusion

In this paper, we focused on the accuracy of IoT device identification, taking into account two critical design factors. Firstly, we utilized device latency instead of deep packet inspection methods due to the latter's high computational demands and privacy concerns. Secondly, we highlighted the significant influence of wireless channel dynamics on device latency features, underscoring the necessity of incorporating these dynamics into the device identification process. To effectively model channel dynamics, we introduced the accumulation score as a metric that is derived from packet capture data and accounts for the impact of wireless channel utilization on the variability of features used in device identification. Our findings reveal that the accumulation score facilitates enhanced data collection for training and also significantly improves the accuracy of device identification when incorporated as an additional feature in training and testing ML algorithms.

Acknowledgment

This work was partially supported by NSF grant #2138633. The authors would like to thank Netgear Inc. (San Jose, CA) for donating some of the materials used to conduct this research.

References

[1] M. R. Santos, R. M. Andrade, D. G. Gomes, and A. C. Callado, "An efficient approach for device identification and traffic classification in iot ecosystems," in *IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2018, pp. 00 304–00 309.

[2] B. Tushir, Y. Dalal, B. Dezfouli, and Y. Liu, "A quantitative study of ddos and e-ddos attacks on wifi smart home devices," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6282–6292, 2020.

[3] A. Osman, A. Wasicek, S. Köpsell, and T. Strufe, "Transparent microsegmentation in smart home iot networks," in *3rd USENIX Workshop on Hot Topics in Edge Computing (HotEdge)*, 2020.

[4] A. Sivanathan, H. H. Gharakheili, and V. Sivaraman, "Detecting behavioral change of iot devices using clustering-based network traffic modeling," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7295–7309, 2020.

[5] 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.

[6] R. R. Chowdhury, S. Aneja, N. Aneja, and E. Abas, "Network traffic analysis based iot device identification," in *Proceedings of the 4th International Conference on Big Data and Internet of Things*, pp. 79–89.

[7] A. Sivanathan, H. H. Gharakheili, F. Loi, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Classifying iot devices in smart environments using network traffic characteristics," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1745–1759, 2018.

[8] V. Thangavelu, D. M. Divakaran, R. Sairam, S. S. Bhunia, and M. Gurusamy, "Deft: A distributed iot fingerprinting technique," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 940–952, 2018.

[9] N. Ammar, L. Noirie, and S. Tixeuil, "Autonomous identification of iot device types based on a supervised classification," in *IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[10] 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 *IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 2177–2184.

[11] A. J. Pinheiro, J. d. M. Bezerra, C. A. Burgardt, and D. R. Campelo, "Identifying iot devices and events based on packet length from encrypted traffic," *Computer Communications*, vol. 144, pp. 8–17, 2019.

[12] H. Gordon, C. Batula, B. Tushir, B. Dezfouli, and Y. Liu, "Securing smart homes via software-defined networking and low-cost traffic classification," in *IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, pp. 1049–1057.

[13] H. Gordon, C. Park, B. Tushir, Y. Liu, and B. Dezfouli, "An efficient sdn architecture for smart home security accelerated by fpga," in *IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*. IEEE, 2021, pp. 1–3.

[14] A. Aksoy and M. H. Gunes, "Automated iot device identification using network traffic," in *IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.

[15] J. Sheth and B. Dezfouli, "Monfi: A tool for high-rate, efficient, and programmable monitoring of wifi devices," in *2021 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2021, pp. 1–7.

[16] J. Sheth, V. Ramanna, and B. Dezfouli, "Flip: A framework for leveraging ebpf to augment wifi access points and investigate network performance," in *Proceedings of the 19th ACM International Symposium on Mobility Management and Wireless Access (MobiWac)*, 2021, pp. 117–125.

[17] Qualcomm. (2020) IPQ8074: High-capacity 802.11ax SoC for Routers, Gateways and Access Points. https://www.qualcomm.com/products/internet-of-things/networking/wi-fi-networks/ipq8074.

[18] E. Valdez, D. Pendarakis, and H. Jamjoom, "How to discover iot devices when network traffic is encrypted," in *IEEE International Congress on Internet of Things (ICIOT)*. IEEE, 2019, pp. 17–24.

[19] A. Aksoy and M. H. Gunes, "Automated iot device identification using network traffic," in *IEEE International Conference on Communications (ICC)*, 2019, pp. 1–7.

[20] F. Shaikh, E. Bou-Harb, J. Crichigno, and N. Ghani, "A machine learning model for classifying unsolicited iot devices by observing network telescopes," in *14th International Wireless Communications & Mobile Computing Conference (IWCMC)*. IEEE, 2018, pp. 938–943.

[21] I. Cvitić, D. Peraković, M. Periša, and B. Gupta, "Ensemble machine learning approach for classification of iot devices in smart home," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 11, pp. 3179–3202, 2021.

[22] O. Salman, I. H. Elhajj, A. Chehab, and A. Kayssi, "A machine learning based framework for iot device identification and abnormal traffic detection," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, p. e3743, 2022.

[23] H. F. Fakhruldeen, M. J. Saadh, S. Khan, N. A. Salim, N. Jhamat, and G. Mustafa, "Enhancing smart home device identification in wifi environments for futuristic smart networks-based iot," *International Journal of Data Science and Analytics*, pp. 1–14, 2024.