PINBALL: Universal and Robust Signature Extraction for Smart Home Devices

Chenxin Duan*, Shize Zhang*, Jiahai Yang*, Zhiliang Wang*, Yang Yang†, Jia Li‡
*Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing, China
*Beijing National Research Center for Information Science and Technology, Beijing, China
†School of Information and Communication, National University of Defence Technology, Xi'an, China
‡National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing, China
Email: {dcx19, zsz16}@mails.tsinghua.edu.cn, {yang, wzl}@cernet.edu.cn, alex_yyy@163.com, lijia@cert.org.cn

***Abstract*—Event-level signatures for smart home Internet-of-Things devices are the fundamental of many security enhancement systems for home networks equipped with smart home systems. However, existing event signature extraction methods for smart home devices are in a dilemma where the precision, generality and robustness of the signatures cannot be satisfied at the same time. In this paper, we present PINBALL, a universal tool that can automatically extract robust signatures for events happening on smart home devices. PINBALL uses length distributions of bidirectional packets generated by smart home devices as signatures for their trigger-events. It is lightweight and independent of the protocols. Besides, the signatures extracted by PINBALL are more robust and achieve better detection accuracy than previous state-of-the-art methods. Nevertheless, we argue that such signatures are double-edged swords that can not only help with the network management but also threaten the user privacy. We analyze these signatures from a comprehensive perspective and try to propose some possible measures to make such packet length side channels able to be used for legitimate purposes like security enhancement with the user privacy preserved.**

***Index Terms*—Internet-of-Things, signature extraction, smart home, user privacy, cyber security**

I. INTRODUCTION

As an application of Internet-of-Things (IoT) technology, smart home ecosystems have been flourishing in recent years because they bring great convenience to people's daily life by enabling the cooperation and remote control of different kinds of smart home devices. However, deployment of smart home devices also poses challenges to the security of the home networks. Due to constrained computation and communication resources, smart home devices behave much differently from those general-purpose devices and are faced with many special security threats [1], [2]. To fortify the security of smart home systems, knowing the profile of the normal behaviors of different smart home devices is the fundamental of many security enhancement measures. In fact, previous works have demonstrated that signatures or fingerprints for events happening on the smart home devices (trigger-events, *e.g.* "toggle ON/OFF" and "Intensity"/"Color") can be extracted based on their traffic characteristics even with the adoption of encryption [3]–[7].

Previous works mainly use two different ways to extract event-level signatures for smart home devices. Some leverage statistical features of the spatial-temporal characteristics of the traffic generated by smart home devices and employ machine learning models to identify which event is happening on the devices that trigger them to generate the traffic [3], [6], [7]. The others try to find some simple but unique features of the events to serve as the signatures that can be identified through direct numerical matching, such as exact values of packet lengths or short sequences of packets with specific lengths and directions [4], [5]. If precise signatures for the communication behaviors can always be got for the smart home devices, network managers will be able to accurately detect the potential intrusions or device misbehaviors in the network that connects these devices. However, given the rapid development and diverse communication techniques of smart home ecosystems, both of these two kinds of methods have limitations. Signatures by machine learning based approaches are not precise enough. Simple but unique signatures lack robustness, because they are sensitive to the disturbance like packet loss, retransmission and out-of-order. Besides, existing methods to extract this kind of signatures are not universal, limited to certain service (web) [4] or protocol (TCP) [5]. These drawbacks may all cause many false positives and false negatives when they are applied to security enhancement in the wild.

To address the problems of existing signature extraction methods for smart home devices, in this paper, we present PINBALL, a universal tool that can extract robust signatures for events happening on smart home devices. PINBALL uses measurable probability distributions of bidirectional packet lengths generated by smart home devices as signatures for the trigger-events. Such signatures can be identified directly by discrepancy metrics and are independent of the communication services or protocols. Thus, PINBALL gains both generality and robustness for the event-level signature extraction of smart home devices.

Despite the advantages of PINBALL, we argue that the event-level signatures for smart home devices extracted by PINBALL and other previous methods are double-edged swords. On one hand, they can be used to monitor the traffic generated by smart home devices and detect potential intrusions and device misbehaviors [3], [8]–[10]. On the other hand, they may cause user privacy leakage because malicious eavesdroppers can use these signatures to infer the activities of the users [6], [7], [11], [12]. For example, if the attackers get the signatures of turning off events for many appliances and the lock event of the home door, they may infer that nobody is in the house now and it is a chance to do something bad. Besides, the signatures for some health-care devices may

978-3-903176-32-4

expose the daily routines or health conditions of the users to undesired strangers. Previous works usually treat these signatures from a one-sided view, either to make use of them to enhance the security or to eliminate them to prevent user privacy leakage. We take the first step to regard this problem comprehensively and discuss how we can utilize these signatures properly. The contribution of this paper can be summarized as follows:

- We propose to use length distributions of bidirectional packets generated by smart home devices as signatures for events happening on them. Experiments show that such probability distribution signatures are more robust to the disturbance and can get better event detection accuracy.
- We design and implement PINBALL, a universal tool that can automatically extract signatures for smart home devices from their traffic traces no matter what protocols they use to communicate.
- We give comprehensive analyses about the implications of the signatures extracted by PINBALL, or at a higher level, the packet-length-based side channels of smart home devices. We try to propose countermeasures to make these signatures able to be used for legitimate purposes such as intrusion detection under the premise that the user privacy will not be exposed.

The remainder of this paper is organized as follows: Section II summarizes the related works. Section III gives an illustrative example for the motivation of PINBALL design and describes the threat model. Section IV demonstrates how PINBALL works. Section V shows the evaluation results. Section VI discusses the countermeasures for the event-level signatures of smart home devices. Section VII concludes the work.

II. RELATED WORK

Device-level Fingerprints: To know what smart home devices are existing in the network is usually the first step for the management of these devices. Therefore, many works try to classify different smart home devices based on their traffic characteristics [13]–[22]. Spatial-temporal features in different resolutions, including packet level, flow level and session level are extracted as fingerprints and then fed into machine learning models to identify the type of the device which generates the traffic. These works only focus on the identification of the device type but do not care about what events happen on the smart home devices. Traffic fingerprints extracted by these works can be regarded as device-level coarse-grained signatures for smart home devices. Nevertheless, only when network managers know the fine-grained event-level signatures of the devices, can they be able to monitor the working states of deployed smart home devices and detect anomalies. Thus, our work concentrates on the more precise signatures that can identify the exact trigger-events of the smart home devices.

Event-level Signatures: HOMESNITCH by OConnor *et. al.* employs statistical features of the throughput, burstiness, synchronicity and duration to identify the activities of smart home devices [3]. Some other works also follow the same way to utilize spatial-temporal features and machine learning models to classify the trigger-events of smart home devices [6], [7], [23]. The statistical features used by these methods can be seen as summaries of many different properties and are not precise enough for missing lots of details. So machine learning models are necessary to find the hidden relations between these features and the trigger-events and help with the identification of the signatures.

Apthorpe *et. al.* show that the traffic rates can be used to infer the changes of the working states of the smart home devices triggered by user actions [11]. Junges *et. al.* demonstrate that the exact values of the lengths of the packets exchanging between the smart home devices and the remote web services can serve as the signatures for the actions performed on the devices [4]. These works use simple values to directly identify the trigger-events of smart home devices. However, they only apply to events which lead to large changes on traffic volume [11] or communications through web services [4].

To our best knowledge, current state-of-the-art method to extract event-level signatures for smart home devices using simple but unique values is PINGPONG presented by Trimananda *et. al.* [5]. They find that just short sequences of packets of specific lengths, exchanged between the devices, the smartphones and the cloud, can uniquely identify the device events. Based on the observation, they design PINGPONG to automatically extract this kind of simple but precise signatures and validate them in many datasets. Nevertheless, PINGPONG also has limitations, with the most notable one being that it only applies to devices that use TCP protocol to communicate. What's more, PINGPONG requires a relatively complex process to extract the signatures, including packet filtering, flow reassembling and clustering. Considering PINGPONG gets the best current performance using simple features directly as event-level signatures for smart home devices, we take it as the baseline of our work and use experiments to show the advantages of PINBALL.

Implications: Some works use event-level signatures of smart home devices as the profile of their normal communication behaviors to monitor the encrypted traffic generated by them and detect potential misbehaviors and intrusions [3], [8]–[10]. At the same time, great concerns have been shown about the threats of these signatures to the user privacy because they can also be used by malicious eavesdroppers to infer the activities of the users. Defensive countermeasures like packet padding [5], [12] and traffic shaping [11] are proposed to prevent this kind of side channel attacks. However, we still lack a comprehensive understanding about the implications of these signatures as double-edged swords. It remains to be investigated whether these signatures can be used for legitimate purposes in a privacy-preserved way.

III. PROBLEM STATEMENT

At first, we use our observation on a concrete smart home device, TP-Link Smart Plug, to illustrate how the design of PINBALL is inspired and how the signatures extracted by PINBALL differentiate from that by previous works. Then, we describe the threat model. Unlike previous works that

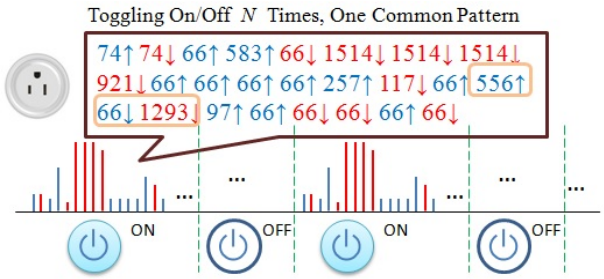

Fig. 1. The sequence pattern of packet lengths and directions for TP-Link Plug toggling on. Every time the device is toggled on, the same packet sequence in a fixed order of packet lengths and directions can be observed.

TABLE I
SIGNATURE BY PINBALL FOR TP-LINK PLUG TOGGLING ON

(l, d)	66 ↑	66 ↓	74 ↑	1514 ↓	**556** ↑	**1293** ↓	...
P	0.277	0.235	0.057	0.051	0.017	0.017	...

only look at this problem from a single perspective, we will consider both a trusted defender who wants to leverage the signatures to enhance the network security and a malicious eavesdropper who attempts to sniff the side channel to access the user privacy.

A. Motivating Example

TP-Link Smart Plug is a typical smart home device that can be turned on and off through the operations on the corresponding application running on a smartphone. When it is toggled on by the local smartphone, although the IP addresses of the remote cloud endpoints vary, a packet sequence with the same pattern of lengths and directions can always be observed, as is shown in Fig. 1 (↑ for sending by the device and ↓ for receiving from the peer, similarly hereinafter). Based on this observation, PINGPONG extracts the length tuple of the "packet pairs" consisting of two ordered packets with TCP payload in opposite directions in the sequence as the signatures for the happening events. Specifically, for the toggling on event of TP-Link Smart Plug, PINGPONG only extracts <556↑, 1293↓> as the signature for this event and the other information in this pattern is discarded based on the design of PINGPONG. This kind of signatures are order-sensitive so that PINGPONG depends on the reliable transmission mechanism of the TCP protocol to recover the possible retransmission and out-of-order packets, leading to that it cannot be applied to devices which only or mainly use UDP protocol to communicate. As for methods that use statistical characteristic values, such as average and maximum, to represent the patterns of packet lengths and directions, more information will be lost and the extracted features on packet lengths will not be distinguishable enough so that additional features like inter-arrival time of the packets and machine learning models are required to improve the event identification accuracy [6], [7].

Patterns with a fixed long sequence of packet lengths and direcitons for a certain event triggering can be observed in many smart home devices. If more information about the sequence patterns rather than only a part of them like "packet pairs" used by PINGPONG can be encoded into the signatures, the extracted signatures will be able to identify the events more accurately. However, it is also expected that the signatures will not be too rigid so that they can be robust for the slight changes of few packets in the signatures caused by the normal updates of the device firmware or disturbance like packet retransmission and out-of-order. Based on these considerations, we propose to use the length distributions of the bidirectional packets generated by the devices after the event triggering as the signatures. The signature for toggling on event of TP-Link Smart Plug extracted by PINBALL is shown in Table I (not complete, the first row is the packet length and direcition tuples and the second is the probability values). Given a large amount of total packets in the duration, the distribution signatures are insensitive to the packet orders and little fluctuations in the number of certain packets. But the occurrence of packets with a key length can be reflected in the sample space of the distribution to make the signatures distinguishable enough. We then design PINBALL to automatically extract this kind of signatures for smart home devices and use them to detect the happening of the events.

B. Threat Model

We consider trusted and malicious sniffers at the same time and describe how they acquire and utilize the signatures. For both of them, we assume that they know the type of the smart home devices they want to monitor but the traffic generated by the devices is all encrypted so that the plaintext communication is invisible. Trusted sniffers know the IP address configurations of all the local smart home devices and can apply tools like PINBALL to extract signatures for their normal behaviors. Then they sniff behind the local gateway router to monitor all the traffic generated by the smart home devices to detect inconsistent behaviors with the known signatures and report them as possible misbehaviors and intrusions. The malicious eavesdroppers can be either a WAN sniffer or a WiFi sniffer. They can acquire the desired signatures of their interested events happening on the smart home devices from another device with the same type. They use the visible IP address fields to distinguish the traffic sources and try to match the signatures with the traffic and then leverage the side channel to infer the user privacy.

The signatures extracted by PINBALL can be used by both the trusted and malicious sniffers. It is expected that the signatures can describe the normal communication patterns of the smart home devices so that they can be used for legitimate purposes like intrusion detection. At the same time, the sensitive information of what really happens on the smart home devices should be hidden by the confusing messages. Proper measures should be taken to meet these two expectations in the future development of smart home IoT ecosystems and we will give a discussion about the possible approaches.

IV. PINBALL DESIGN

PINBALL has two working phases: training phase and detection phase. In the training phase, PINBALL tries to extract signatures for the target event from labelled traffic traces. In

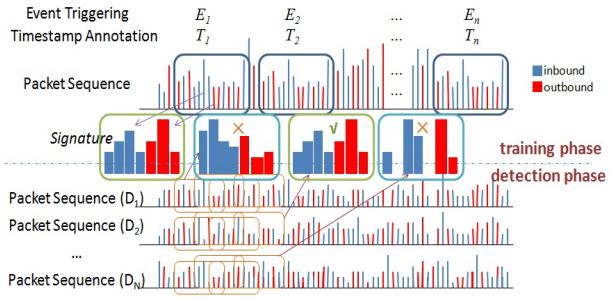

Fig. 2. Overview of PINBALL

the detection phase, PINBALL monitors all the traffic generated by different devices in the network and detects traces which can match the known signatures. For illustrative purpose, we describe the working processes of PINBALL from a local WiFi sniffer's perspective. That is to say, PINBALL can use IP addresses to distinguish traffic generated by different devices. In this case, PINBALL extracts event signatures from all the traffic generated by the smart home devices in a period without caring about the communication objects and protocols. As for a WAN sniffer who may monitor the traffic in front of a NAT gateway with all the traffic multiplexed onto the router's IP address, PINBALL can also extract flow-level event signatures for smart home devices by aggregating traffic with the same 5-tuple flow identifier (source and destination IP address, source and destination port and layer 4 protocol). In fact, for many devices, the packets in their event signatures come from a single flow. The overview of PINBALL is shown in Fig. 2 and we then describe how PINBALL works in details.

A. Training Phase

In the training phase, PINBALL processes traffic traces generated by a certain smart home device under the force of a series of target event triggering. To collect the training data for PINBALL, a total of N events (the same event type, *e.g.* toggling on, repeated for N times), denoted as $\{E_1, E_2, ..., E_n\}$, should be manually triggered with long enough time intervals. And the traffic traces in this period are captured with the timestamps of each event triggering annotated as $\{T_1, T_2, ..., T_n\}$. For each triggered event E_i, PINBALL only focuses on the packets generated in a t-second short time window after the event triggering and gets a subset of packets P_i in which the time of all the packets generated by the target device is in $[T_i, T_i + t]$. The length of the time windows, t, is not a sensitive parameter and can be set as 10 seconds to be a proper value, which is long enough to allow all traffic related to the event to complete. PINBALL is only interested in the lengths and directions of the packets in $\bigcup\{P_1, P_2, ..., P_n\}$ and regards them as a random variable, whose value is a 2-tuple that represents the length and direction of a packet like $(length, direction)$. The direction can be inbound or outbound and the packet lengths are constrained by the minimum frame size and the maximum transmission unit of the data link layer. The output of PINBALL in training phase is a discrete probability distribution $P(X)$ and it serves as the final signature for the corresponding event.

PINBALL will select length and direction tuples of packets that are strongly relevant with the triggered event to be added into the sample space of $P(X)$. For the determination of the packets strongly relevant with the triggered event, PINBALL finds the packets whose length and direction tuples occur for more than $0.9N$ times after a total of N events (the performance is also not very sensitive to the factor 0.9 and there is a trade-off between the precision and robustness). In other words, for a length and direction tuple denoted as (l, d), if there are packets coming from more than $0.9N$ packet sets among a total of N packet sets $\{P_1, P_2, ..., P_n\}$ collected from the triggered events, with their length and direction tuple the same as (l, d), then the tuple (l, d) will be added into the sample space of the final distribution signature $P(X)$. Sometimes, packets in the same direction and simliar lengths may occur in turns after the different triggering of the same event. Therefore, if packets in the same direction whose lengths are in a continuous range also occur for more than $0.9N$ times after a total of N events, a range of packet length and direction tuples, denoted as $([l_a, l_b], d)$, can also be selected as a whole to be added into the sample space of the final distribution signature. Note that it is required that every single tuple (l, d) with $l_a \leq l \leq l_b$ must occur at least once in the full set $\bigcup\{P_1, P_2, ..., P_n\}$ and otherwise tuples cannot be aggregated as a continuous range with packets in some certain lengths missed.

After the determination of the sample space of $P(X)$, denoted as $X = \{(l, d)_1, ...(l, d)_i, ...([l_{ja}, l_{jb}], d_j), ...\}$, we calculate their probabilities as frequencies: $P(x) = Count(x) / \sum_{x_i \in X} Count(x_i)$, where $Count(\cdot)$ is a function that takes a length and direction tuple (l, d) as input and returns the number of packets whose length and direction tuple are the same as (l, d) in the full set $\bigcup\{P_1, P_2, ..., P_n\}$. Finally, we will get a discrete probability distribution describing a random variable whose value is either an exact tuple of packet length and direction or a continuous range of packet lengths with the same direction. A distribution in the following format is the final signature extracted by PINBALL.

$$\begin{bmatrix} X \\ P(X) \end{bmatrix} = \begin{bmatrix} ... & (l, d)_i & ... & ([l_{ja}, l_{jb}], d_j) & ... \\ ... & p_i & ... & p_j & ... \end{bmatrix}$$

B. Detection Phase

In the detection phase, PINBALL monitors all the traffic generated by different devices in the network and tries to find traffic that matches the known signature and reports it as a detection of the event. Supposing PINBALL is detecting an event whose signature is a discrete probability distribution with a k-dimensional sample space, denoted as $P(X) = (p_1, p_2, ..., p_k)$, PINBALL sets a sliding time window in the same length as that in the training phase (t) and calculates the distribution of the k values in the sample space of $P(X)$ in every time window by estimating probability with frequency for packets generated by every device in the network. The sliding time window moves in a stride of 1 second. Thus, for every second and every device in the network, PINBALL will get a

distribution with the same sample space as $P(X)$, denoted as $Q(X) = (q_1, q_2, ..., q_k)$. PINBALL determines whether $Q(X)$ is a match with the signature $P(X)$ by calculating discrepancy metrics for them and if their discrepancy metrics are less than the thresholds, PINBALL will report that a corresponding event of $P(X)$ happens on the device which generates the distribution $Q(X)$ at that time.

Metrics that measure the discrepancy of two probability distributions can be used by PINBALL to determine whether the distribution in a time window $Q(X)$ is a match with the signature $P(X)$. Hellinger distance and KL-divergence are both typical metrics that measure the discrepancy of two probability distributions, with their formulation being $H(P,Q) = \frac{1}{\sqrt{2}}\sqrt{\sum_{i=1}^{k}(\sqrt{p_i} - \sqrt{q_i})^2} = \frac{1}{\sqrt{2}}||\sqrt{P} - \sqrt{Q}||_2$ and $D_{KL}(P||Q) = \sum_{i=1}^{k} p_i \log \frac{p_i}{q_i}$. Hellinger distance is the probabilistic analog of Euclidean distance with a bounded range [0, 1]. KL-divergence is also called relative entropy and is widely used in machine learning fields to measure the discrepancy of two probability distributions. These two metrics are both employed by PINBALL. However, notice that if some tuples with low probabilities in the signature $P(X)$ are missed with a 0 probability in $Q(X)$, this kind of discrepancy cannot be well reflected in the metric of Hellinger distance and KL-divergence. But some key packet length and direction tuples with low probabilities in the signatures also matter for the determination of the signature match. The occurrence of some key packets should be necessary even though they may have low probabilities in the signatures. Therefore, we define the third metric called occurrence discrepancy in the following formulation:

$$f(p) = \begin{cases} \frac{1}{p} & \text{p} \geq 0.01 \\ 100 & \text{otherwise} \end{cases}, OD(P,Q) = \frac{\sum_{j}^{q_j=0} f(p_j)}{\sum_{i}^{k} f(p_i)}.$$

In this metric, only values in the sample space of $P(X)$ that are missed in $Q(X)$ contribute to the discrepancy value. The function $f(p)$ calculates the weights of different values in this metric based on their probabilities in the signature $P(X)$. And the smaller the probability of the value in $P(X)$ is, the higher weight it gets in this metric. A bound of 0.01 is set to avoid some values getting too high weights. Values with high probabilities in the signature only get little weight in this metric because a large value of Hellinger distance and KL-divergence metrics will be caused once they are missed.

A practical way to decide the thresholds for the metrics is to set them according to the discrepancy values between the signatures and the distributions calculated from the time windows after $\{T_1, T_2, ..., T_n\}$ in the training phase. The thresholds for match with the signatures can be set as a value that is a little bit greater than the maximum metric values observed on the groundtruth events in the training phase. In fact, as will be demonstrated later, benefiting from the discrimination of the signatures extracted by PINBALL, it is okay to set the thresholds in a relatively wide range.

Finally, PINBALL adds a debounce handling to the report of the detected events. Because the packets generated after an event triggering can span a long duration but the sliding window only moves in the stride of 1 second, there may be several successive matches with the signatures in a series of time windows. PINBALL only reports the detected event once for these successive matches.

V. EVALUATION

PINBALL is evaluated on a wide range of public available datasets covering lots of different smart home devices. Our prototype implementation of PINBALL and signatures extracted by PINBALL on the evaluation datasets are also released for public access [24]. The datasets used to evaluate PINBALL include:

Dataset 1: PingPong Evaluation Dataset. This dataset [5] is built by the authors of PINGPONG to evaluate the performance of PINGPONG. Traffic traces of 19 popular smart home devices are collected in the dataset. The dataset is made up of a training set and a testing set. In every subset, a total of 100 events are generated for each device with the timestamps automatically recorded (for events with binary values, *e.g.* toggling on and toggling off, each of them are generated 50 times and there are totally 25 different event types for all the devices). In the training set, the traffic of each device is collected individually without any background traffic. And in the testing set, traffic generated by different devices are mixed up with additional background traffic generated by other IoT devices and general-purpose devices. As PINBALL takes the same data format as inputs as PINGPONG, this dataset can be directly used for the evaluation of PINBALL. We apply PINBALL to extract signatures and detect events for the smart home devices in this dataset from a WiFi sniffer's perspective and compare the results with PINGPONG. However, due to the limitation of PINGPONG, the devices under evaluation in this dataset all use TCP protocol to communicate.

Dataset 2: MonIoTr Dataset. The MonIoTr dataset [25] contains traffic generated by 55 distinct smart home devices in different locations and network settings. For every device in the dataset, different events are triggered manually for several times and the traffic generated in a short period after the event triggering is collected with the timestamps of event triggering annotated. We apply PINBALL to extract event signatures for the devices in the dataset. A set of devices which mainly use UDP protocol to communicate are included in the dataset and can be used to test the generality of PINBALL by checking whether it can successfully extract event signatures for devices using UDP protocol.

Dataset 3: UNSW Dataset. The UNSW dataset [15], [21] is collected from a real-life "smart environment" that serves a wide range of IoT and non-IoT devices over its network infrastructure. There are traffic traces generated by 24 smart home devices and 6 general-purpose devices during a period of 20 days in the dataset.

Dataset 4: Yourthings Dataset. The Yourthings dataset [1] contains traffic traces for 45 smart home devices collected in a period of 10 days.

Dataset 5: CICIDS2017 Dataset. This dataset contains simulated network traffic for an office space with two servers

TABLE II
EVALUATION RESULTS AND COMPARISON BETWEEN PINBALL AND PINGPONG

	match rate (per 100 events, dataset 1)	false positive rate (per 100 events, dataset 1)	UDP support (dataset 2)	false positives (per signature, dataset 3, 4 and 5)
PINBALL	**98.40**	**0.08**	✔	**3**
PINGPONG	97.56	0.32	✘	11

TABLE III
SIGNATURE BY PINBALL FOR D-LINK PLUG STATE CHANGE

(l,d)	54 ↑	66 ↑	74 ↑	288 ↑	**784** ↑	**1052** ↑	54 ↓
P	0.263	0.132	0.050	0.031	0.031	0.030	0.180
(l,d)	66 ↓	74 ↓	**91** ↓	103 ↓	**647** ↓	**1227** ↓	
P	0.124	0.031	0.033	0.032	0.032	0.032	

and 10 laptops/desktops with diverse operating systems [26]. We only use the Monday trace of normal traffic without any injected intrusion traffic in this dataset.

PINBALL cannot extract signatures for the devices in dataset 3, 4 and 5 due to the lack of the annotations about the event triggering. However, the traffic traces in these 3 datasets are generated by different smart home devices and general-purpose devices that are not contained by dataset 1. Thus, we use these datasets to check the uniqueness of the signatures extracted by PINBALL by detecting events in these datasets with the signatures extracted from the PingPong evaluation dataset, with the expectation that no event (false positive) should be detected in these 3 datasets. It is also very important to evaluate whether the signatures are unique enough to avoid causing large amounts of false positives when they are used to detect events in the wild.

The evaluation results and comparison between PINBALL and PINGPONG are summarized in Table II. Firstly, for all the devices and events that PINGPONG can extract signatures for, PINBALL can also extract signatures for them. And in the event detection evaluation, PINBALL achieves higher detection rate for events that really happened on the devices and lower false positive rate than PINGPONG. Besides, for a set of devices that mainly use UDP protocol to communicate and PINGPONG cannot be applied to, including Ring Doorbell, Bosiwo Camera and Lightify Hub, *etc.*, PINBALL also successfully extracts event signatures for them. Finally, in the uniqueness checking evaluation, despite large volumes of traffic data in the 3 datasets containing more than 400 million packets in total, the signatures extracted by PINBALL are sufficiently unique to avoid too many false positives. PINBALL also reports fewer false positives in these 3 datasets than PINGPONG because more information about the communication patterns of the events is encoded into the signatures extracted by PINBALL.

We then use 3 concrete devices in the PingPong evaluation dataset to illustrate how the signatures extracted by PINBALL look like and why PINBALL can achieve better performance than PINGPONG.

D-Link Plug State Change: The signatures for the state change event (switch between power on and off) of the D-Link Plug device extracted by PINGPONG from the communication between the device and the remote cloud endpoints are two

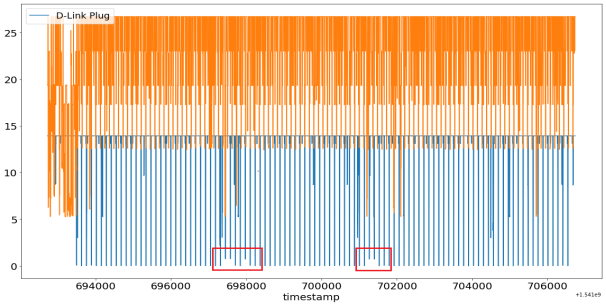

Fig. 3. KL-divergence metrics for D-Link Plug State Change

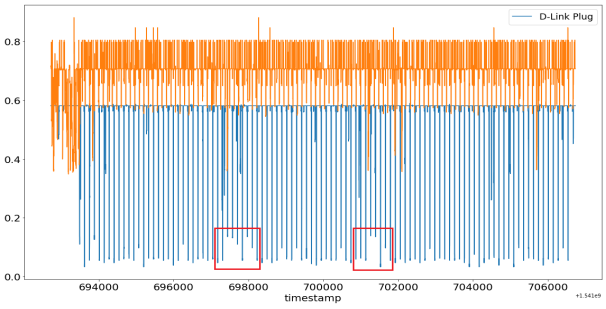

Fig. 4. Hellinger distance metrics for D-Link Plug State Change

short sequences of packet lengths and directions: <91↓, 1227↓, 784↑> and <1052↑, 647↓>. The packet length distribution signature extracted by PINBALL for the same event is shown in Table III. The length and direction tuples in the signature extracted by PINBALL all come from a single TCP flow and we can see that the key packet length and direction tuples in the signatures extracted by PINGPONG are all included in the signature by PINBALL. The metric values of KL-divergence and Hellinger distance in the event detection process of PINBALL are shown in Fig. 3 and Fig. 4 (The other line represents the discrepancy metric values of another device in the dataset that are smallest among the devices except the D-Link Plug itself and the metric values of other devices are always too large so that are omitted in the figures). We can see that in the time windows after the triggered events, the discrepancy metric values are almost 0 between the signature and the corresponding distribution of packet lengths and directions of the device. And the threshold values for the metrics can be set in a relatively wide range. Besides, PINGPONG can only detect 95 matches with the signature for the events that are actually triggered 100 times but PINBALL can perfectly detect all of them. This may result from the facts indicated by the areas in the red rectangles in the figures. We can observe slight increases in the discrepancy metric values for some triggered events caused by the packet losses. Benifiting from more information encoded into the signature, the lost packets only have a very small impact on the event detection of PINBALL and it can still detect them with the threshold values set properly. However, for PINGPONG, once the key packets are lost, the signatuers will be invalid and PINGPONG will miss these actually triggered events.

SmartThings Plug Toggling On/Off: SmartThings Plug is a smart home device that works relying on the Samsung SmartThings programmable IoT platform [27]. We use this

TABLE IV
SIGNATURES BY PINBALL FOR SMARTTHINGS PLUG

(a) Toggling On

(l, d)	66 ↑	74 ↑	**136** ↑	279 ↑	**699** ↑
P	0.310	0.050	0.104	0.049	0.053
(l, d)	66 ↓	74 ↓	**511** ↓	*612* ↓	**777** ↓
P	0.229	0.050	0.049	0.049	0.058

(b) Toggling Off

(l, d)	66 ↑	74 ↑	**136** ↑	279 ↑	**700** ↑
P	0.319	0.052	0.097	0.047	0.049
(l, d)	66 ↓	74 ↓	**511** ↓	*616* ↓	**780** ↓
P	0.233	0.053	0.048	0.047	0.054

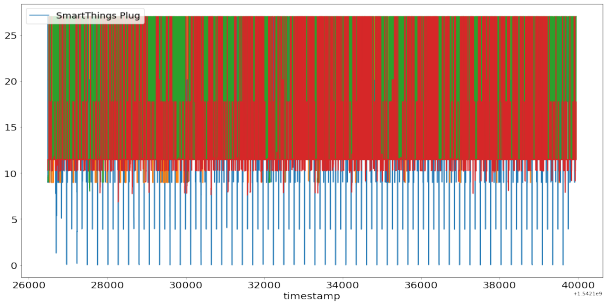

Fig. 5. KL-divergence metrics for SmartThings Plug Toggling On

TABLE V
SIGNATURES BY PINBALL FOR NEST THERMOSTAT

(a) Fan On

(l, d)	66 ↑	74 ↑	66 ↓	74 ↓
P	0.464	0.055	0.323	0.055
(l, d)	$[891, 894]$ ↑		$[830, 834]$ ↓	
P	0.052		0.052	

(b) Fan Off

(l, d)	66 ↑	74 ↑	66 ↓	74 ↓
P	0.481	0.055	0.317	0.059
(l, d)	$[858, 860]$ ↑		$[829, 834]$ ↓	
P	0.044		0.045	

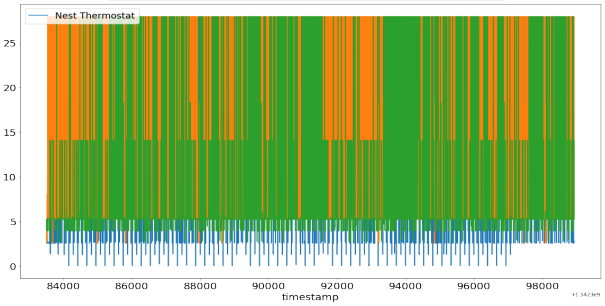

Fig. 6. KL-divergence metrics for Nest Thermostat Fan On

case to illustrate the differences in the signatures of dual events extracted by PINBALL for the same device (*e.g.* toggling on and off). The signature extracted by PINGPONG for toggling on event of the SmartThings Plug from the communication between the local smartphone and the remote cloud endpoints is <699↑, 511↓> and <777↓, 136↑> and that for toggling off event is <700↑, 511↓> and <780↓, 136↑>. The signatures extracted by PINBALL for these two dual events are shown in Table IV. Again, we can see that the packet length tuples in the signatures extracted by PINGPONG are all included in the signatures by PINBALL. Besides, PINBALL finds additional packets that can distinguish these two events (612↓ for toggling on and 616↓ for toggling off). The KL-divergence metrics (Hellinger distance is similar) in the detection process of toggling on event by PINBALL are shown in Fig. 5 (The toggling on and toggling off events are triggered alternately for 50 times each and some other devices that have a relatively small metrics are also shown in the figure, similarly hereinafter). Although there are many common packet length and direction tuples with similar probabilities in the signatures of these two dual events, the different packets in the signatures are still distinguishable enough and the threshold values can also be set in a relatively wide range. For these two events, PINGPONG only detects 92 matches with the signatures, however, PINBALL detects all the actually triggered events again without any miss. We think this can be partially attributed to the common packet length and direction tuples in the two signatures. Although they cannot distinguish these two dual events, they help with the detection of the happening of both the two events. And it is unwise for PINGPONG to discard this kind of imformation in the signatures.

Nest Thermostat Fan On/Off: We use this case to show the effect of the third discrepancy metric called occurrence discrepancy used in PINBALL. The signatures extracted by PINGPONG for these two events of Nest Thermostat are <[891, 894]↑, [830, 834]↓> for the fan on event and <[858, 860]↑, [829, 834]↓> for the fan off event. The signatures extracted by PINBALL for these two events are shown in Table V. We can see that PINBALL finds the same ranges of packet lengths with PINGPONG in the signatures. Besides the two continuous ranges of packet lengths in the signatures, the other packet length and direction tuples are common in the signatures of other devices and have high probabilities (packets in 66-byte and 74-byte length are usually TCP headers plus different optional fields without any payload). And the KL-divergence metrics and occurrence discrepancy metrics in the detection process of the fan on event by PINBALL are shown in Fig. 6 and Fig. 7. We can see that the KL-divergence metrics are not as distinguishable in this case as it is in the former 2 cases. There are only small differences in KL-divergence metrics between the dual on/off events and the distributions generated by other devices. However, the occurrence discrepancy metrics have high distinguishability for the signatures of these two events because the range values with low probabilities in these two signatures can get high weights in this metric. For these two events, PINGPONG can only detect 92 matches with one false positive and PINBALL detects 95 matches without false positives. For some events that both PINBALL and PINGPONG fail to detect, it may be due to the fact that too many distinguishable packets in the signatures are lost (see the areas in the red rectangles indicating large occurrence discrepancy values in Fig. 7). The same reason may also account for why PINBALL still has some misses for the detection of the actually triggered events.

To sum up, the evaluation results show that PINBALL is a universal tool that can extract event signatures for smart home devices no matter what protocols they use to communicate. Moreover, the signatures extracted by PINBALL are also more

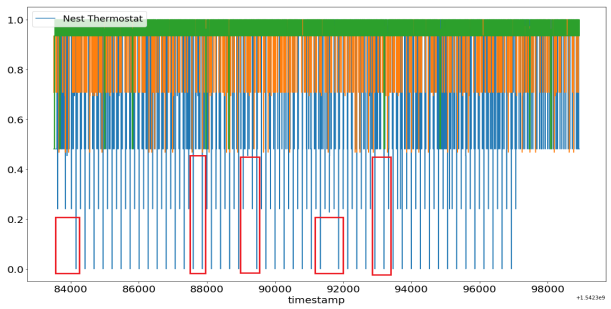

Fig. 7. occurrence discrepancy metrics for Nest Thermostat Fan On

robust because PINBALL has more information about the communication patterns encoded into the signatures in order-insensitive probability distribution formats. Thus, PINBALL can achieve higher event detection rate and lower false positive rate than pervious state-of-the-art methods.

VI. POSSIBLE COUNTERMEASURES

Despite the generality and robustness of PINBALL, such signatures are double-edged swords. Considering the threat to the user privacy, many previous works try to propose methods to eliminate this packet-length-based side channel, including packet padding, traffic shaping and traffic injection [5], [11], [12], [28]. It is suggested in the discussion about PINGPONG that a middlebox-based VPN gateway that pads all the packets generated by the smart home devices to the same length (*e.g.* MTU) is a feasible approach to defend the side channel attacks [12], [28]. However, the VPN-based technique may work for those devices that only or mainly use TCP protocol to communicate, but for devices that PINGPONG does not apply to and use UDP protocol to exchange large volumes of real-time data, like IP cameras and digital video recorders (DVR), a middlebox-based VPN will cause high overhead and it may inhibit the normal functionalities of these devices by causing long transferring latency. Moreover, The middlebox-based approach only defends against the malicious WAN sniffers except the local WiFi sniffers and it is impractical to implement on-device VPN due to the constraint of the limited computational resources of the smart home devices. Last but not the least, complete elimination of these side-channels will also make the security enhancement systems that monitor the traffic generated by the devices based on the packet length signatures fail to work [8]–[10].

Regarding the signatures by PINBALL as double-edged swords, we really need a trade-off between data availability and data privacy. This problem is also investigated by the differential privacy community [29]. Inspired by this, we propose to use techniques in differential privacy area to deal with the event signatures of smart home devices. More concretely, we can use mechanisms like randomized response (coin flipping) [30] to add noise to these signatures. A possible approach that can satisfy both sides may be to pad the packets in a regular way and generate additional confusing traffic that can hide what exactly happens on the smart home devices but also provide a clear profile about the normal communication behaviors of the devices. For example, the same sequence pattern can be used for the toggling on and toggling off events for smart plug devices. As can be observed in the existing smart plug devices, there are only small differences in the sequence patterns of the toggling on and toggling off events of the same device so it will not take much effort to pad them into the same pattern. If the events of toggling on and off have the same signature, the malicious eavesdroppers will not be able to distinguish which event is happening on the device. Nevertheless, this may be not enough because it will fail once the adversaries get to know only one exact event that happens and then they can infer the other events in the sequences based on the paired occurrence of the two events. Thus, it is suggested that the devices can generate traffic in the same patterns without the corresponding events really triggered in their idle periods to confuse the adversaries so that an observed signature does not always indicate a real event happening. That is to say, we break the mapping relationship between the traffic signatures and the trigger-events of the smart home devices rather than eliminate the signatures. In this way, the intrusions that do not exhibit these patterns, like scanning and flooding attacks, can also be detected based on the signatures and the generated decoy traffic in idle periods will not inhibit the functionalities of the devices.

The implementations of the smart home devices need to be changed to adopt the meatures we propose above. As for devices that have been deployed, this can be finished by the firmware update. In fact, privacy-by-design framework that gives guidance on the development and implementation of smart home devices has already been proposed [31]. But previous works that focus on the user privacy issues of the smart home ecosystems concentrate more on what data the devices collect and how they are stored and uesd [31]–[33], the side channels that may also expose the user privacy have been ignored. We leave the empirical evaluation of our proposed countermeasures in the future work and we also hope other effective methods that make full use of the packet-length-based signatures we present in this paper as a double-edged sword can be proposed.

VII. CONCLUSION

In this paper, we present PINBALL, a universal tool that can automatically extract robust signatures for smart home devices. We use experiments to show that PINBALL is a universal application and has advantages on simplicity and robustness compared with previous methods. We also take the first step to regard the packet-length-based side channels of smart home devices in a comprehensive perspective and propose possible measures that can be taken to make the event signatures of these devices able to be used by trusted network managers in a privacy-preserved way.

ACKNOWLEDGMENT

This work is supported by the National Key Research and Development Program of China (No.2017YFB0803004). Yang Yang is supported by the Research Program of National University of Defence Technology (No.ZK18-03-59).

REFERENCES

[1] O. Alrawi, C. Lever, M. Antonakakis, and F. Monrose, "Sok: Security evaluation of home-based iot deployments," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, pp. 1362–1380.

[2] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis *et al.*, "Understanding the mirai botnet," in *26th {USENIX} security symposium ({USENIX} Security 17)*, 2017, pp. 1093–1110.

[3] T. OConnor, R. Mohamed, M. Miettinen, W. Enck, B. Reaves, and A.-R. Sadeghi, "Homesnitch: behavior transparency and control for smart home iot devices," in *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*, 2019, pp. 128–138.

[4] P. Junges, J. François, and O. Festor, "Passive inference of user actions through iot gateway encrypted traffic analysis," in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2019, pp. 7–12.

[5] R. Trimananda, J. Varmarken, A. Markopoulou, and B. Demsky, "Packet-level signatures for smart home devices," in *Network and Distributed System Security Symposium, NDSS*, 2020.

[6] A. Acar, H. Fereidooni, T. Abera, A. K. Sikder, M. Miettinen, H. Aksu, M. Conti, A.-R. Sadeghi, and S. Uluagac, "Peek-a-boo: I see your smart home activities, even encrypted!" in *Proceedings of the 13th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2020, pp. 207–218.

[7] B. Copos, K. Levitt, M. Bishop, and J. Rowe, "Is anybody home? inferring activity from smart home network traffic," in *2016 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2016, pp. 245–251.

[8] W. Zhang, Y. Meng, Y. Liu, X. Zhang, Y. Zhang, and H. Zhu, "Homonit: Monitoring smart home apps from encrypted traffic," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 1074–1088.

[9] L. Cheng, K. Tian, and D. Yao, "Orpheus: Enforcing cyber-physical execution semantics to defend against data-oriented attacks," in *Proceedings of the 33rd Annual Computer Security Applications Conference*, 2017, pp. 315–326.

[10] T. Gu, Z. Fang, A. Abhishek, H. Fu, P. Hu, and P. Mohapatra, "Iotgaze: Iot security enforcement via wireless context analysis," in *IEEE Conference on Computer Communications, IEEE INFOCOM 2020*, 2020.

[11] N. Apthorpe, D. Reisman, S. Sundaresan, A. Narayanan, and N. Feamster, "Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic," *arXiv preprint arXiv:1708.05044*, 2017.

[12] A. J. Pinheiro, J. M. Bezerra, and D. R. Campelo, "Packet padding for improving privacy in consumer iot," in *2018 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 2018, pp. 00 925–00 929.

[13] Y. Meidan, M. Bohadana, A. Shabtai, J. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici, "Profiliot: a machine learning approach for iot device identification based on network traffic analysis," in *SAC 2017: The 32nd ACM Symposium On Applied Computing*, 2017, pp. 506–509.

[14] M. Miettinen, S. Marchal, I. Hafeez, T. Frassetto, N. Asokan, A. Sadeghi, and S. Tarkoma, "Iot sentinel demo: Automated device-type identification for security enforcement in iot," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 2511–2514.

[15] 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, 2019.

[16] Y. Meidan, M. Bohadana, A. Shabtai, M. Ochoa, N. O. Tippenhauer, J. Guarnizo, and Y. Elovici, "Detection of unauthorized iot devices using machine learning techniques," *arXiv preprint arXiv:1709.04647*, 2017.

[17] B. Bezawada, M. Bachani, J. Peterson, H. Shirazi, I. Ray, and I. Ray, "Iotsense: Behavioral fingerprinting of iot devices." *arXiv preprint arXiv:1804.03852*, 2018.

[18] J. Ortiz, C. H. Crawford, and F. Le, "Devicemien: network device behavior modeling for identifying unknown iot devices," in *Proceedings of the International Conference on Internet of Things Design and Implementation, IoTDI'19*, 2019, pp. 106–117.

[19] M. R. Shahid, G. Blanc, Z. Zhang, and H. Debar, "Iot devices recognition through network traffic analysis," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 5187–5192.

[20] S. Marchal, M. Miettinen, T. D. Nguyen, A. 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.

[21] A. Sivanathan, D. Sherratt, H. H. Gharakheili, A. Radford, C. Wijenayake, A. Vishwanath, and V. Sivaraman, "Characterizing and classifying iot traffic in smart cities and campuses," in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2017, pp. 559–564.

[22] M. Xiaobo, Q. Jian, L. Jianfeng, L. John C.S, L. Zhenhua, and G. Xiaohong, "Pinpointing hidden iot devices via spatial-temporal traffic fingerprinting," in *IEEE Conference on Computer Communications, IEEE INFOCOM 2020*, 2020.

[23] B. Charyyev and M. H. Gunes, "Iot event classification based on network traffic," in *IEEE INFOCOM Workshops*, 2020.

[24] Pinball: Universal and Robust Signature Extraction for Smart Home IoT Devices. [Online]. Available: https://github.com/ZebornDuan/Pinball

[25] J. Ren, D. J. Dubois, D. Choffnes, A. M. Mandalari, R. Kolcun, and H. Haddadi, "Information exposure from consumer iot devices: A multidimensional, network-informed measurement approach," in *Proceedings of the Internet Measurement Conference*, 2019, pp. 267–279.

[26] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." in *ICISSP*, 2018, pp. 108–116.

[27] Samsung SmartThings. Add a little smartness to your things. [Online]. Available: https://www.smartthings.com.

[28] N. Apthorpe, D. Y. Huang, D. Reisman, A. Narayanan, and N. Feamster, "Keeping the smart home private with smart(er) iot traffic shaping," *privacy enhancing technologies*, vol. 2019, no. 3, pp. 128–148, 2019.

[29] C. Dwork, "Differential privacy: A survey of results," in *International conference on theory and applications of models of computation*. Springer, 2008, pp. 1–19.

[30] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, vol. 60, no. 309, pp. 63–69, 1965.

[31] C. Perera, C. McCormick, A. K. Bandara, B. A. Price, and B. Nuseibeh, "Privacy-by-design framework for assessing internet of things applications and platforms," in *Proceedings of the 6th International Conference on the Internet of Things*, 2016, pp. 83–92.

[32] M. Ghiglieri and E. Tews, "A privacy protection system for hbbtv in smart tvs," in *2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)*. IEEE, 2014, pp. 357–362.

[33] T. S. Saponas, J. Lester, C. Hartung, S. Agarwal, T. Kohno *et al.*, "Devices that tell on you: Privacy trends in consumer ubiquitous computing." in *USENIX Security Symposium*, 2007, pp. 55–70.