Peek-a-Boo: I see your smart home activities, even encrypted!

Abbas Acar[1], Hossein Fereidooni[2], Tigist Abera[2], Amit Kumar Sikder[1], Markus Miettinen[2], Hidayet Aksu[1], Mauro Conti[3], Ahmad-Reza Sadeghi[2], Selcuk Uluagac[1]
[1]Florida International University - {aacar001,asikd003,haksu,suluagac}@fiu.edu
[2]TU Darmstadt - {hossein.fereidooni,tigist.abera,markus.miettinen,ahmad.sadeghi}@fiu.edu
[3]University of Padua - conti@math.unipd.it

ABSTRACT

A myriad of IoT devices such as bulbs, switches, speakers in a smart home environment allow users to easily control the physical world around them and facilitate their living styles through the sensors already embedded in these devices. Sensor data contains a lot of sensitive information about the user and devices. However, an attacker inside or near a smart home environment can potentially exploit the innate wireless medium used by these devices to exfiltrate sensitive information from the encrypted payload (i.e., sensor data) about the users and their activities, invading user privacy. With this in mind, in this work, we introduce a novel *multi-stage privacy attack* against user privacy in a smart environment. It is realized utilizing state-of-the-art machine-learning approaches for detecting and identifying the types of IoT devices, their states, and ongoing user activities in a cascading style by only passively sniffing the network traffic from smart home devices and sensors. The attack effectively works on both encrypted and unencrypted communications. We evaluate the efficiency of the attack with real measurements from an extensive set of popular off-the-shelf smart home IoT devices utilizing a set of diverse network protocols like WiFi, ZigBee, and BLE. Our results show that an adversary passively sniffing the traffic can achieve very high accuracy (above 90%) in identifying the state and actions of targeted smart home devices and their users. To protect against this privacy leakage, we also propose a countermeasure based on generating spoofed traffic to hide the device states and demonstrate that it provides better protection than existing solutions.

CCS CONCEPTS

• **Networks → Network privacy and anonymity**; Security protocols; • **Hardware → Analysis and design of emerging devices and systems**; • **Security and privacy →** *Mobile and wireless security*.

KEYWORDS

Privacy; Smart-home; Network Traffic; WiFi; ZigBee; BLE

ACM Reference Format:
Abbas Acar[1], Hossein Fereidooni[2], Tigist Abera[2], Amit Kumar Sikder[1], Markus Miettinen[2], Hidayet Aksu[1], Mauro Conti[3], Ahmad-Reza Sadeghi[2], Selcuk Uluagac[1]. 2020. Peek-a-Boo: I see your smart home activities, even encrypted!. In *13th ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec '20), July 8–10, 2020, Linz (Virtual Event), Austria.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3395351.3399421

1 INTRODUCTION

Previously, the Internet was mainly used for accessing and displaying content of web pages (i.e., web browsing). However, with the emergence of IoT devices in smart homes, users have now the ability to control their home's electronic systems (e.g., smart bulbs, smart locks, sensors) using appropriate smartphone apps and also from remote locations [37, 42]. To realize smart home automation, the devices are mostly equipped with embedded sensors. These sensors collect data from the environment and help users to control them. Moreover, smart home devices are also continuously communicating with associated back-end system servers or other devices (e.g., smart hubs) to transmit the sensor data in a real-time manner. On the other hand, as IoT devices usually are single-purpose devices, the capabilities of individual smart home devices are relatively limited, comprising only a few states or actions. For example, a motion sensor allows a user to detect any movement in a physical space, but the sensor has only two states: motion and no-motion. If an attacker can reveal the current state of the sensor, the attacker will also reveal the presence of the user at home.

Given that the communications among the server, sensors, smart-hub, and the smart home devices are usually encrypted using standard protocols like WPA2, in the case of WiFi, the contents of the exchanged messages or commands are hidden. However, the encryption only hides the payload, related meta-data (e.g., packet lengths, traffic rate) of the network traffic still leaks some information about the messages exchanged [11, 13, 25, 44, 45, 48].

Identification of the encrypted traffic is a well-studied problem. However, applying traditional identification methods such as statistical techniques [45] in the domain of smart home is not straightforward due to challenges arising from the inherent properties of IoT devices. First, unlike targets using a widely-deployed protocol to perform a well-known specific activity like web browsing, in the smart home context, the targeted device population is much more heterogeneous and uses various network protocols such as WiFi, ZigBee, BLE, etc. for supporting an even wider variety of device-type-specific, potentially proprietary application protocols. This naturally extends the potential attack surface, but also makes it even harder to devise *generic* attacks or countermeasures.

Some earlier works [4, 5, 43] have shown that it is relatively easy to make some simple inferences such as device type inference [28], identifying the user occupancy via detecting the mode transition between the device activities [14], or simple device mode inference [5]. However, combining such partial information from different smart home devices to get a more meaningful picture about a user's actions or his/her activity profile is challenging. This is because a successful attacker must aggregate information about actions over a longer period of time from a multitude of smart home devices, which is only feasible if activity detection and identification can be *automated* to a large degree to keep the required effort manageable.

In this paper, we demonstrate how machine learning methods based on traffic profiling of smart home IoT device communications can be used by an adversary to automatically identify actions and activities of the IoT devices and its users in a victim's smart home with very high accuracy, even if only encrypted data are available. Indeed, device types, daily mundane activities of the users (e.g., left home, walking from kitchen to bedroom), or states of the devices (e.g., door locked, unlocked) can all be easily identified even if the traffic is encrypted, thus posing a threat to user privacy. We refer to this novel attack to user privacy as *multi-stage privacy attack*, which is achieved in a cascading style by only observing passively the wireless traffic from smart home devices. In this, a passive attacker can easily realize the multi-stage privacy attack to extract meaningful data from any smart environment equipped with smart devices including personal homes, residences, hotel rooms, offices of corporations or government agencies. Here, unlike earlier approaches, the presented attack is device-type and protocol-agnostic, making it easily applicable to a wide variety of different IoT device types without the need for tedious harvesting of device-type or protocol-specific knowledge about specifications for supporting the activity identification task.

We evaluate the effectiveness of the novel multi-stage privacy attack with 22 different off-the-shelf IoT devices utilizing the most popular wireless protocols for IoT. Our experimental results show that an attacker can achieve very high accuracy (above 90 %) in identification of the types, actions, states, activities of the devices and sensors. Moreover, to counter the identified privacy threats posed by the multi-stage privacy attack, we also propose a new effective countermeasure solution based on generating spoofed traffic to hide the real states of targeted IoT devices and thereby the real activities of the users. Our solution does not require modifications in targeted IoT devices and is, therefore, easier to deploy than previously proposed solutions for IoT devices, for which it is very difficult to implement client-based countermeasures due to the vast heterogeneity of smart devices and limited resources available on the IoT devices. Also, even if the user is not at home, a fake traffic-based solution for the user's presence will mask the user's absence, further improving privacy.

Contributions : The contributions of this work are as follows:

- We propose a *novel multi-stage privacy attack* on smart home users and devices which can leak sensitive information including types of devices, states of the devices, sensor data, and on-going user activities. To the best of our knowledge, this is the first work showing all the stages of an attack that can reveal the user activities from the raw traffic of smart home devices, even encrypted.
- The attack includes several novel techniques, both theoretical and practical, for effectively reducing the effort needed for the user activity inference on the timing-based heterogeneous network traffic. First, we demonstrate how to convert the user activity inference on the timing-based heterogeneous network traffic into a cascaded Machine Learning (ML) problem. Then, we further extend the attack by modelling the user activities via the Hidden Markov Model (HMM).
- We evaluate our proposed novel attack with a dataset of 22 popular commercial smart home devices. We show that an attacker can automatically detect and identify device actions with high accuracy ($>$ 90%), allowing an adversary to infer potentially sensitive information about the smart home users.
- Finally, although the focus of this paper is on the novel attack, we also propose a new solution based on traffic spoofing to address this new privacy threat while demonstrating its efficacy over existing solutions (Sec. 5).

Organization: The rest of this paper is organized as follows: In Section 2, we present the adversary model considered in this paper. Then, a background about the communication features of the smart home devices are presented in Section 3. Section 4 details the stages of our main proposed multi-stage attacks, where the results are presented in every sub-section. In addition, we present a solution to mitigate this privacy leaks in Section 5 and we discuss some related issues in Section 6. Finally, the related work are given in Section 7 and the paper is concluded in Section 8.

2 ADVERSARY MODEL

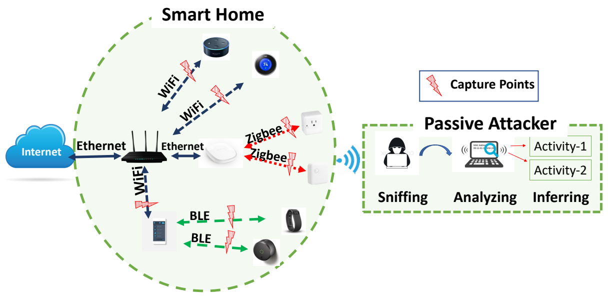


Figure 1: Local adversary model considered in this paper.

One of the unique challenges in the domain of IoT, and particularly smart home, is that the attack surface is naturally extended and comprises a diverse set of devices and sensors deployed at the user's home. Figure 1 shows some of the data capturing points that an attacker can take advantage of when inferring user activities. In this work, we consider a local adversary located physically within the wireless range of the targeted user's smart home devices similar to [20–22]. For this, the attacker can install the sniffers only once and even manage them remotely. Or, it could compromise a device inside the smart home, remotely, and turn it into a sniffer. In this way, the attacker may never need to be present. In all these cases, the adversary can eavesdrop on various wireless IoT network communications transmitted by the user's smart home devices. For example, as presented in Figure 1, the attacker can sniff all the network traffic transmitted over WiFi, BLE, and ZigBee protocols. The attacker only needs to passively sniff the network traffic and does not need to interrupt. Therefore, the attacker may stay active long enough without being detected by the victim. An

Table 1: The communication protocols and capabilities of the smart home devices used.

		Communication			Capabilities		
ID	Device	WiFi	ZigBee	BLE	Type-I	Type-II	Type-III
1	ApexisCam	●	○	○	○	○	●
2	AirRouter	●	○	○	●	○	○
3	AugustSmartlock	○	○	●	○	●	●
4	BelkinWemoLink	○	○	○	○	●	○
5	DLinkCam	●	○	○	○	○	●
6	DLinkDoorSensor	●	○	○	○	○	●
7	DLinkMotionSensor	●	○	○	○	○	●
8	DLinkSiren	●	○	○	○	○	●
9	EdimaxCam	●	○	○	○	○	●
10	EdimaxSPlug1101	●	○	○	○	●	○
11	EdinetCam1	●	○	○	○	○	●
12	EdinetGateway	●	○	○	●	○	○
13	FitbitAria	●	○	○	○	●	○
14	Lightify2	●	○	○	○	●	○
15	PhilipsHueBridge	●	○	○	●	○	○
16	SMCRouter	●	○	○	●	○	○
17	STMotionSensor	○	●	○	○	○	●
18	STOutlet	○	●	○	●	●	○
19	STMultiSensor	○	●	○	○	○	●
20	TPLinkHS110	●	○	○	○	●	○
21	WansviewCam	●	○	○	○	○	●
22	WemoInsightSwitch	●	○	○	○	●	○

Type-I: Hub-like devices, Type-II: User-controlled devices, Type-III: Sensor-like devices

alternative adversary would be an adversary who can launch the attack remotely, i.e., intercepting the network traffic over the Internet such as a malicious ISP. We further discuss the advantages and limitations of such an adversary in Section 6.

Assumptions. We further make the following assumptions:

- The attacker has access to the same kind of smart home devices and sensors as the targeted user, s/he can analyze the devices by collecting the traffic of these devices, and use the collected data to train its algorithms.
- The attacker has access to protocol headers data on all layers that are not protected by encryption. The attacker does not need to know the specifications of analyzed protocols, instead it only needs to know how to run the already publicly available scripts, which does not require an extensive knowledge about the specifications of the protocol itself. Moreover, it can also use Layer 2 information like MAC addresses, or BLE advertisement packets, to automatically identify additional information, the brand of individual devices, thereby reducing the search space of devices to guess the set of smart home devices that the targeted user is using. Moreover, it is also worth noting that the attacker does not need exactly same devices to train its algorithm, but it needs exact brand and device type to get the results presented in this paper as we use the $< brand, device - type >$ pair to uniquely identify devices.

3 SMART HOME DEVICES

In this section, we present the background information of the communication protocols used by the smart devices.

3.1 Communication Features

Both the smart home vendors and users mostly prefer wireless communication over wired communication as it is more convenient. However, compared to wired communication, the wireless network traffic from smart home devices is open to the eavesdropping attacks.

In this work, we target three wireless protocols: WiFi, ZigBee, and Bluetooth Low Energy (BLE). Table 1 shows all the devices studied in this paper Table 1 with their wireless communication protocol and device capabilities. Among these protocols, WiFi is used in the wired or plugged-in devices, while other protocols, ZigBee and BLE, are implemented for short range communication tasks of battery-powered devices as they consume less power than WiFi.

3.1.1 WiFi-enabled devices. WiFi-enabled devices are connected to the Internet either through a Hub-like device or directly connected to an access point. In both cases, the adversary can track and capture the traffic through a specific device via MAC address. Even though MAC addresses may help the attacker to narrow down the device type, it can not precisely decide the device type from MAC address. It may want to use IP addresses of servers. However, the adversary can only see the traffic that is encrypted by both the network protocols (SSL/TLS) and WiFi encryption (WPA). Therefore, it cannot see the IP or transport layer headers encrypted by the WPA protocol. This prevents the attacker from using header-based features for the device identification. However, the traffic rates of the devices still cannot be hidden from the attacker.

3.1.2 ZigBee-enabled devices. ZigBee devices have two addresses: MAC address and Network Address (NwkAddr). The MAC address is exactly the same as the MAC used in WiFi-enabled devices, which is unique for every device in the world and never changes. On the other hand, NwkAddr is created and assigned when the device joins a network and changes when it leaves and re-joins another network. It is similar to IP, however, it is not encrypted and source and destination NwkAddr of the packets can be seen by the attacker. In addition, the network coordinator (i.e., hub) has the $0x0000$ address and each network has a unique identifier, called the Personal Area Network Identifier (PAN ID). This information may additionally help the attacker.

3.1.3 BLE-enabled devices. In a BLE network, a device can be either a master or a slave. A slave can connect to only one master node while a master can connect to multiple slave nodes. In all the smart home devices that we used, while the smartphone acts as a master, targeted smart device acted as a slave. Before establishing the connection, a slave device broadcasts advertising packets (ADV_IND) randomly on channel 37, 38, and 39. Once a connection starts, they agree on a channel map, where they follow in the rest of the communication. If an attacker wants to follow the BLE traffic through a smart device, it needs to capture the first packet so that it can learn the channel mapping. Once the attacker captures the access address, it can follow the rest of the communication.

4 MULTI-STAGE PRIVACY ATTACK

As shown in Figure 2, our novel multi-stage privacy attack consists of four stages connected in a cascaded manner. While the goal of

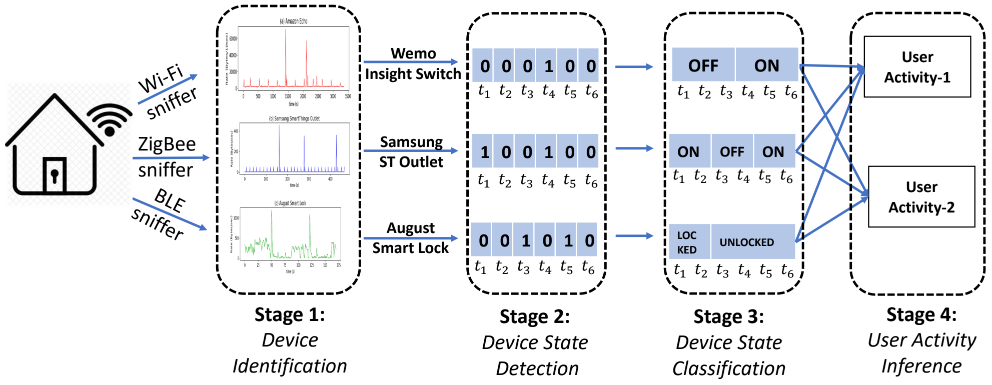


Figure 2: Overview of our multi-stage privacy attack.

the attack is to infer user activities at the final stage, every stage also leaks partial information about devices and their actions and can be independently used by the attacker for various purposes. Before going into further details about our attack, in Appendix B, we show the feasibility and possibility of privacy leaks from encrypted network traffic of smart home devices. Particularly, we show that an attacker who can sniff the network traffic of the devices can easily infer some simple information without using any advanced techniques. We consider one device for each protocol: Wemo Insight Switch (WiFi), Samsung ST Outlet (ZigBee), and August Smart Lock (BLE). We analyze the raw network traffic of each device and see if it is really possible to extract information from the network traffic, specifically from data rate. In the next sections, we describe the details of individual stages and evaluate the efficiency of our multi-stage privacy attack on network traffic data collected from 22 different off-the-shelf IoT devices used in smart homes.

4.1 Dataset and Evaluation Metrics

In order to evaluate the attacks in the stages above, we collected the network data from 22 different smart home devices. Data collection was performed in two stages: In the first stage, controlled experiments were performed in which detailed instructions were followed to initiate specific actions on the tested device. These instructions were compiled based on the on-line or hardcopy manual of each tested device (specs and data sheets). The controlled experiments were performed in order to ensure that all relevant actions for each device were represented in the usage dataset sufficiently many times. Each experiment was therefore repeated $n = 20$ times for each device. In addition to the controlled experiments, also uncontrolled testing was performed in order to capture background traffic of relevant devices. In this set-up, several devices were configured to be used simultaneously and device actions were occasionally triggered during a test period of ca. 1-2 hours.

The duration and the total size of the captures and the number of the packets are given in Table 2. The devices used include a representative cross-section of IoT device types, typically available in the European and North American markets during the study. The devices were also selected based on the market share of different device categories. The most popular device categories are smart security systems such as smart cameras and smart locks (22.2%), lighting (3.03%), outlets and switches (1%), gateways including hubs and routers (24.5%), and smart speakers (22.39%) [2]. In addition to

Table 2: Characteristics of network traces used in experiments.

Device	Period (mins)	Size (MB)	Packets
ApexisCam	133	80	152220
AirRouter	85	49	115192
AugustSmartLock	25.8	0.66	8129
BelkinWemoLink	71	0.66	2039
DLinkCam	225	1.15	5389
DLinkDoorSensor	74	0.48	3519
DLinkMotionSensor	74	0.47	2849
DLinkSiren	71	0.41	3073
EdimaxCam	225	0.27	1798
EdimaxSPlug1101	74	0.5	2823
EdinetCam1	117	0.3	2779
EdinetGateway	225	0.34	3240
FitbitAria	213	0.043	257
Lightify2	74	0.25	1022
PhilipsHueBridge	53	0.8	2680
SMCRouter	124	47	150768
STOutlet	6	0.04	1061
STMotionSensor	11	0.05	1291
STMultiPurpose	12	0.22	5255
TPLinkHS110	71	0.14	473
WansviewCam	193	11	73759
WemoInsightSwitch	117	0.8	1675

these categories, we also included several smart sensors as these devices hold significant smart home market share (approximately 23.9%) [23]. We installed all the devices in a laboratory network and emulated user inputs triggering device state changes. We captured all the network traffic from a device and performed the analysis offline.

For evaluating the efficiency of our attacks, we use different metrics. First, we use accuracy, which is the ratio of correctly inferred observations to total observations. In some cases, as in real deployments, the collected network data may have imbalanced data, where the duration of the active state is much less than the inactive one. In those cases, we use additional metrics such as Precision, Recall, F1 score, and Support. In the cases that the dataset includes a lot more label 0 (no activity) rows than label 1 (activity) rows, we observed that F1 score is a better performance measurement than accuracy although accuracy is a more intuitive performance measurement, in general. The detailed calculation of these evaluation metrics is given in Appendix A.

4.2 Calculating Features from Network traffic

In this sub-section, we explain how we use the traffic flow for the classification task. Particularly, we take advantage of the fact that while the encryption layer in the protocol protects the payload of a packet, it fails to hide other information revealed by network traffic patterns, for instance, sequence of packet lengths (SPL) and direction (incoming/outgoing). We consider each network traffic flow as a time ordered sequence of packets exchanged between two peers during a session. Before processing the network traffic for classification, we converted packet in traffic flow into a Sequence of Packet Lengths and Times (SPLT) as in following format:

$$pkt = [timestamp,\ direction,\ packet\ length], \tag{1}$$

where the direction is 1(0) if it is an incoming (outgoing) packet. This transformation is done for each packet in the captured trace, where each result is written to a new row. In the end, we obtained a matrix with three columns. Then, in the feature extraction of each attack, we calculated the features from this matrix.

4.3 Stage-1: Device Identification

Several different identification approaches for IoT devices have been proposed in literature. Numerous works have shown that IoT devices can be identified with high accuracy for both WiFi-enabled [7, 15, 27, 28, 30] and BLE-enabled [16] devices. Therefore, in this section (e.g., Stage-1), we implemented already existing device identification algorithm for ZigBee-enabled smart home devices using our features to see whether we can identify the ZigBee-enabled smart home devices from their network traffic.

In our dataset, each device can be uniquely identified by the $< brand, device - type >$ pair. We did not consider the different models of devices as different devices. On the other hand, a hub in ZigBee always uses the network address $0x0000$, so it can be easily recognized by the attacker. Therefore, we did not include the hub in the identification of ZigBee devices. After collecting ZigBee network traffic, the second step involves extracting the features to identify the devices. In this step, the features we used include *mean packet length*, *mean inter-arrival time*, and *standard deviation in packet lengths*. We split each individual network traffic trace of a device into equal time intervals (e.g., 5 sec, 10 sec). Then, we calculated these features for each interval.

For the classification, we used the kNN classification algorithm. The classifier could correctly identify devices with an overall accuracy of 93% for ZigBee devices. This shows that as for WiFi and BLE, also devices using ZigBee can be identified with high accuracy.

4.4 Stage-2: Device State Detection

When an interaction between the device and the user occurs, a significant amount of data is transmitted, which leads to a significant increase in the traffic rate. After this data exchange, the data transmission drops to the minimum until a new interaction starts. When there is no activity, only the minimum amount of continuation packets like heartbeat messages are sent to minimize the device's power and bandwidth consumption. We also observed that almost the same amount of data transfer occurs for the same activities. All this information allows us to detect transitions between the activities or states of the device. For further validation, we do the following experiments.

Table 3: Evaluation results of device activity detection stage.

Device	**Random Forest**		**kNN**	
	F1 Score	Accuracy	F1 Score	Accuracy
ApexisCam	93	97	94	98
AirRouter	98	97	98	97
AugustSmartLock	100	100	100	100
BelkinWemoLink	80	79	85	83
DLinkCam	85	80	85	80
DLinkDoorSensor	94	98	92	97
DlinkMotionSensor	74	96	69	95
DlinkSiren	89	99	91	99
EdimaxCam	84	82	82	81
EdimaxSPlug1101	91	97	92	97
EdinetCam1	76	96	76	96
EdinetGateway	80	99	66	99
FitbitAria	100	100	100	100
Lightify2	86	99	81	98
PhilipsHueBridge	74	98	76	98
SMCRouter	94	91	100	100
STOutlet	83	99	92	99
STMotionSensor	91	97	92	97
STMultiSensor	86	99	92	99
TPLinkPlug1101	98	99	92	99
WansviewCam	91	87	91	86
WemoInsightSwitch	86	98	88	98
Avg	**88**	**99**	**91**	**95**

4.4.1 Feature Extraction. Our goal is to transform a sequence of packets into a supervised learning dataset. To achieve this, we divided the sequence of packets into windows of size W. For a given time interval length W, we extracted a feature vector comprised of three variables: *mean packet length*, *mean inter-arrival time* and *median absolute deviation of packet size*. Based on timestamped labels telling whether an activity was ongoing or not, we labeled the given vector with 1 for an ongoing activity or 0 for no activity. We found that the window size has significant influence on the performance of our model. The window size for the best performance depends on adjusting the size according to the duration of the activity. In general, selecting a smaller window size improves the performance until some level, but any further reduction results in decline of the performance. From our observation, better performance was observed when the window size is about a quarter of the duration of an activity.

4.4.2 Results. After obtaining feature vectors with labels from the sequence of packets, any supervised learning algorithm can be applied on the dataset. We have evaluated two supervised learning algorithms, namely Random Forest classifier (RF) and k-Nearest Neighbors classifier (kNN). As shown in Table 3 both RF and kNN have similar performance with RF averaging 88% and kNN with 91% average of correctly detecting activities. F1 Score of each device in Table 3 differs slightly. DlinkMotionSensor has the worst F1 score 74% using RF and 69% using kNN and the best F1 score is 100% for the Aria Fitbit and AugustSmartLock.

4.5 Stage-3: Device State Classification

In the device state classification experiments, the attacker's goal is to decide the state of the device (e.g., deciding if it is ON or OFF). When looking at the device's exchanged network packets, unlike previous steps, this is more difficult to determine. However, each state has a unique pattern which helps us to differentiate them from each other. In order to see if it is possible to differentiate the states, we did the following experiments:

4.5.1 Feature extraction. To conduct device state classification, informative and distinctive features must be extracted from time-series generated in the preprocessing steps. We used the *tsfresh* [12] tool that automatically calculates a large number of time series characteristics and features and then constructed our feature vector. Examples of the features extracted from time-series are as follows: Absolute Energy of time-series, Length of time-series, Mean and median of time-series, Skewness of time-series, Entropy of time-series, Standard deviation of time-series, Variance of time-series, Continuous wavelet transform coefficients, Fast Fourier Transform Coefficients, Coefficients of polynomial fitted to time-series.

4.5.2 Feature selection. The output of the feature extraction phase is a set of feature vectors including 795 binary features. A large number of features, some of which redundant or irrelevant might present several problems such as misleading the learning algorithm, and increasing model complexity. A feature selection technique was therefore used to mitigate these problems and also to reduce over-fitting, training time and improve accuracy. We used a technique leveraging ensembles of randomized decision trees (i.e., Extra Trees-Classifier) for determining the importance of individual features. We exploited Extra-Trees Classifier to compute the relative importance of each attribute to inform feature selection. The features considered unimportant were discarded. The feature selection phase effectively reduced the feature vector size from 795 to 197 binary features.

4.5.3 Results. Our objective was to build a performant model to correctly classify IoT devices' states even if their traffic is encrypted. To this end, we employed several machine learning algorithms for the classification such as *XGBoost*, *Adaboost*, *Random Forest*, *SVM with RBF kernel*, *kNN*, *Logistic Regression*, *Naïve Bayes*, and *Decision Tree*. In order to ensure that our machine learning model got the most of the patterns from the training data correctly, and it was not picking up too much noise, we shuffled and split the data-points to conduct the following experiments: *(i)* we performed 5-fold Cross Validation (CV) on a training set of 377 samples (75% of data) for assessing the effectiveness of the machine learning model and *(ii)* we carried out Hold-out Validation on 126 samples (25% of data) to test the machine learning model performance against unseen data.

5-fold Cross Validation: To avoid the risk of missing important patterns or trends in the dataset, we applied cross validation, as it provides ample data for training the model and also leaves ample data for validation. Thus, we conducted a 5-fold cross validation experiment. In 5-fold CV, the data are randomly partitioned into 5 equal-sized sub-samples. Of the 5 sub-samples, a single sub-sample is retained as the validation data for testing the model, and the remaining 4 sub-samples are used as training data. The process is then repeated 5 times with each of the 5 sub-samples used exactly once as the validation data. The 5 results from the folds can then be averaged to produce a single estimation. We obtained 92% accuracy in terms of F1 Score in the detection of devices' states using Random Forest classifier, as shown in Table 4.

Table 4: Cross-validation and hold-out validation results for device state classification.

Classifier	**5-fold CV**	**Held-out data** (25% of data)		
	(75% of data)	Precision	Recall	F1 Score
SVC RBF Kernel	86	89	87	87
Logistic Reg.	87	90	89	88
Random Forest	**92**	96	94	**94**
Naive Bayes	87	92	87	88
Decision Tree	66	62	63	61
K-NN	84	91	87	87
Adaboost	86	89	87	87
XGBoost	85	91	87	87

Table 5: Hold-out validation results of RF classifier for all IoT devices.

Device name	**Action**	**Pre.**	**Recall**	**F1**	**Supp.**
ApexisCamera	live view	100	100	100	4
AirRouter	surfing on amazon	80	100	89	4
AugustSmartLock	off	100	67	80	3
AugustSmartLock	on	67	100	80	2
BelkinWemoLink	off	80	100	89	8
BelkinWemoLink	on	100	50	67	4
DLinkCamera	live view	100	100	100	3
DLinkDoorSensor	open	100	100	100	5
DLinkSensor	motion detection	100	100	100	6
DLinkSiren	turn on	100	100	100	1
EdimaxCam	live view	100	100	100	1
EdimaxSPlug1101	on	100	100	100	5
EdinetCam1	live view	100	100	100	2
EdinetGateway	on	100	100	100	3
FitbitAria	measure weight	100	100	100	4
Lightify2	change light type	100	100	100	6
PhilipsHueBridge	turn scene off	100	100	100	3
PhilipsHueBridge	turn scene on	100	100	100	5
SMCRouter	surfing on amazon	100	80	89	5
STOutlet	on	100	89	94	9
STMotion	active	88	100	93	7
STMotion	inactive	100	71	83	7
STMultiSensor	acceleration active	100	100	100	8
STMultiSensor	acceleration inactive	71	100	83	5
TPLinkPlugHS110	turn off	100	100	100	5
WansviewCam	reboot	100	100	100	9
WemoInsightSwitch	on	100	100	100	2
Avg./Total	————	**96**	**94**	**94**	**126**

Hold-out Validation: To make sure that our classifier can generalize well and is not over-fitted, we tested the classifiers' performance in terms of Precision, Recall, and F1 Score against unseen data (the data was removed from the training set and is only used for this purpose). Table 5 shows the detailed results obtained by Random Forest classification algorithm when conducting the device state classification over 126 unseen samples. As can be seen, the F1 Score of each device used in the experiment differs slightly. We obtained an average performance measurement of 0.94 (94%) of correctly classifying activities. This shows that an attacker can easily differentiate the devices' states.

4.6 Stage-4: User Activity Inference

Modern smart home environments comprise several sensors and devices that are connected with each other and share information. These devices and sensors are configured as independent entities, but work co-dependently to provide an autonomous system. Any

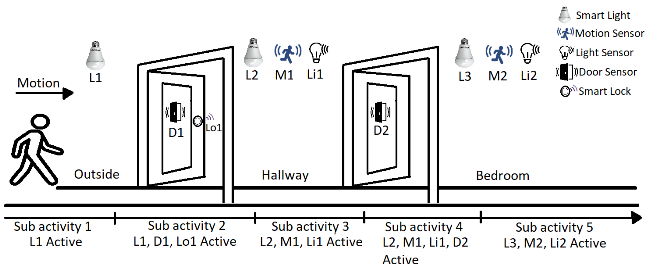


Figure 3: User walking scenario in a smart home.

user activity in a smart home can be inferred by observing the states of the devices and sensors [38, 39].

4.6.1 Modelling User Activities. In Figure 3, we demonstrate a simple walking scenario of a user. Here, a user is entering the smart home from outside to the bedroom through the hallway. The scenario consists of five different devices with lights both inside and outside the home controlled by the motion sensor (M) and light sensor (L). This simple activity can be illustrated as a sequential pattern: Sub-activity 1- moving towards the door from outside (L1 is active), Sub-activity 2- user opens the front door (L1, D1, Lo1 are active), Sub-activity 3- user enters the hallway (L2, M1, Li1 are active), Sub-activity 4- user enters the room (Li2, L2, M2, D1, Lo1 are active), Sub-activity 5- user inside the home (L2, M2, Li2 are active). To complete the activity, a user must follow the same sequence of sub-activities and complete each step. Here, the devices' states (active/inactive) for a specific time can be determined from the network traffic captured from the devices. These device states can be used to infer an on-going activity in a smart home setting.

4.6.2 Feature Extraction. To infer user activities, different device features must be extracted from network traffic data. Network traffic data contain several features including timing information, sensor information, device states, location, etc. Based on the data-type, the extracted features from the network traffic for user activity inference can be represented as follows:

$$Data\ array, E_T = \{S, D, M, L\}, \tag{2}$$

where T is the timing features extracted from the network traffic, S is the set of sensors' features, D is the set of device features, M is the features extracted from the controlling device (smartphone/tablet), and L is the set of location features extracted from the network traffic. We describe the characteristics of these features below.

- *Timing features (T):* Smart home devices change their state according to user activities and commands. Some devices perform time-independent tasks (e.g., switching lights with motion), while some devices perform a task in a certain pattern with different user activities (e.g., walking from one point to another) based on smart home settings. We extract the time of an event from the network traffic captured from different devices to build the overall state of the smart home at the time of the user activity.
- *Sensor State features (S):* Smart home environment consists of different sensors (e.g., motion sensor, light sensor, door sensor, etc.) which act as a bridge between devices and the peripheral. Sensors in a smart home can sense different environment parameters which can trigger different pre-defined tasks in multiple devices. Moreover, sensors can sense any change occurred because of a user interaction and forward this information as an input to the associated devices. These sensor data can be both logical (motion sensor) and numerical (temperature sensor) depending on the nature of the sensor. We observe the changes in both logical and numerical value of a sensor from the captured network traffic and use as a feature to infer user activities. We represent the changes in sensor data as binary output: 1 for active state and 0 for inactive state.
- *Device State features (D):* In a smart home environment, multiple devices such as smart light, smart thermostat, etc. can be connected with each other and with a central hub to perform different tasks. These devices can be configured to change their states (active/inactive) to perform a pre-defined task or to perform a task based on user activities. We consider the state information of all the connected devices as features and extract this information from captured network traffic to infer the on-going user activity. The active and inactive states of the devices are illustrated as 1 and 0 respectively in the data array.
- *Controller State features (M):* Smart home devices can be controlled in an autonomous way and also by using a controller device (smartphone/tablet). To understand the changes in states of the sensors and devices, one should consider the control commands generated by the controller devices. We consider the state of controller device as active (represented as 1 in data array) when a user interacts with smart home devices via controller device and inactive otherwise (represented as 0 in data array). This state information of the controller devices can be extracted from the captured network traffic to build the data array.
- *Controller Location features (L):* The devices connected in a smart environment can be controlled from a different location and this location information can be collected from the captured network traffic. We consider the location of the controller device as a feature to understand any activities on smart home. We consider the home location of the controller device as 1 and the away location of the controller as 0 to represent the location feature as a binary number in the data array.

For Stage 4, we captured the network traffic from a smart home environment and create the feature array explained in Equation 2. We captured the network traffic for a specific time to correctly portray user activities from the network data. Each element of the data array represents the operating conditions of different smart devices, sensors, and controller devices. These data were then used to train a Hidden Markov Model (HMM) to detect user activities in a smart home environment.

To train this HMM, we collected data from a smart home environment with real smart devices. We consider common smart home devices to build our training environment [17]. Our test smart home environment included Samsung SmartThings hub, Samsung multipurpose sensor, Samsung motion sensor, Netgear Arlo security camera, Philips Hue smart light, Ecobee Smart Thermostat, and August Smart Lock. We collected network traffic data from 10 different users for different user activities.

4.6.3 Activity Types. User activities in a smart home environment can be instantaneous (e.g., switching on a device) or sequential over time (e.g., walking from one place to another). We categorized user activities in a smart home environment in two categories - time-independent and time-dependent user activities.

Table 6: Typical user activities in a smart home.

Task Category	Task Name
Time-independent	1. Controlling device within smart home.
	2. Controlling device from outside of the home.
	3. Presence in a specific point at home.
Time-dependent	4. Walking in the smart home.
	5. Opening/ closing doors/windows.
	6. Entering/ exiting from smart home

- *Time-independent Activities:* These user activities are instantaneous, non-sequential activities which do not depend on time. For example, a user can switch on/off a device in the smart home environment at a specific time instance. This activity will show changes in different features for only one time.
- *Time-dependent Activities:* These user activities are time-dependent, sequential activities. For example, a user can move from one point to another point. This activity will show changes in different features over time in a specific sequence.

We tested our HMM model with data collected from six different user activities. We selected these activities as these are the common user activities listed in prior works [40]. As we intend to make our activity detection generalized, we do not consider any rare events that has less possibility of happening in a real-life environment. Our user activity model is explained below.

- *User Activity- 1.* A user is controlling a device from inside of the smart home environment.
- *User Activity- 2.* A user is controlling a device from outside of the smart home environment.
- *User Activity- 3.* A user is performing tasks from a specific point of a smart home environment.
- *User Activity- 4.* A user is walking from one point to another inside the smart home environment.
- *User Activity- 5.* A user is entering/ exiting from the smart home environment.
- *User Activity- 6.* A user is opening/ closing a window/ door in smart home environment.

4.6.4 Results. To train our proposed HMM for user activity inference, we collected user activity data for a week from 15 different people (total 30 datasets) in an emulated smart home environment. We asked the users to perform their daily activities in a timely manner (from morning to night) and performed the same activities in defined sequences in a real-life smart home setting. We considered single authorized smart home user interacting with smart devices at a time for data collection. We trained our HMM model with these data. We also collected data for this activity model to test our proposed method. We collected two datasets for each activity (12 in total) to test the efficacy of the activity inference model.

In Table 7, the evaluation results of our activity inference model are shown. For time-independent activities (Activity-1, Activity-2, and Activity-3), one can infer with 100% accuracy and F-score from the captured network traffic data in a smart home environment. On the contrary, accuracy and F-score decreases slightly for time-dependent activities as these activities introduce FP and FN instances in the activity inference model. For Activity-4, our proposed stage 4 activity inference HMM can achieve both accuracy and F-score over 95%. The false positive rate (FPR) and false negative rate (FNR) are over 5% and 3% respectively for Activity-4. For Activity-4 and Activity-5, the accuracy of user activity inference decreases (93% and 94% respectively) while FPR and FNR increases. The reason for the increment of FPR and FNR is that different time-dependent user activities can have similar patterns over time with small changes in specific time instances. This affects the probability of occurring an activity calculated from HMM. In summary, an attacker can infer time-independent activities more accurately (with 100% accuracy and F-score) than the time-dependent activities (with over 95% accuracy and F-score).

Table 7: User activity inference from network traffic data in a smart home environment.

Smart Home User Activity	TPR	FNR	TNR	FPR	Accuracy	F-score
Activity-1	1	0	1	0	1	1
Activity-2	1	0	1	0	1	1
Activity-3	1	0	1	0	1	1
Activity-4	0.96	0.03	0.94	0.05	0.95	0.95
Activity-5	0.95	0.04	0.87	0.12	0.93	0.91
Activity-6	0.97	0.02	0.91	0.08	0.94	0.94

Finally, note that an accurate user activity inference means that all the stages in the multi-stage attack have to be correctly guessed, which may lower the end-to-end successful inference rate of the attacker. For example, if the stage 1, 2, 3, and 4 are X, Y, Z, and T, respectively, for an attacker, the probability of correctly guessing the Activity-4 of the user is $X \times Y \times Z \times T$. However, we also note that independently inferred information in every stage is also valuable as it may also include sensitive information (e.g., inferring the device type of a connected medical device may reveal the health status of the subject [29, 46]).

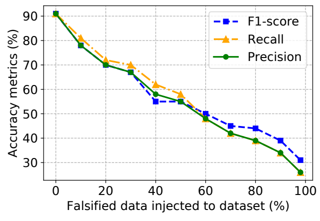
5 MITIGATING THE PRIVACY LEAKS

Despite the security vulnerabilities exploited before, as these privacy concerns are *inherent* and *insidious*, it is too hard to detect and avoid these types of threats associated with smart home devices. An attacker can passively listen to the wireless medium and record all the network traffic from a smart home environment without interrupting the normal activities of devices and their users.

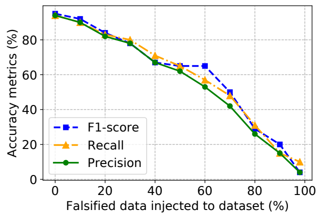
5.1 Proposed Approach

In this sub-section, we propose a solution based on generating spoofed traffic. In this way, even if the user is not at home, generating false activity for the user's presence traffic will mask the user's absence.

In order to measure the efficacy of our proposed spoofed traffic, we investigated the injection of false packets by modifying the feature vectors and evaluated how the performance measurements would change. Then, we applied it to the device state detection and device activity classification attacks. Since the user activity inference is based on the results of the device state detection and device activity classification attacks, if we can falsify their results, the attacker will not able to infer the activities correctly. Particularly, we conducted a set of experiments where we injected falsified data into the training set to observe how the previously shown detection and classification algorithms would behave in such a situation. The results are shown in Figure 4.



(a) Device state detection



(b) Device state classification

Figure 4: Impact of false data injection experiments on the attack accuracy.

Impact of False Data Injection on Device State Detection. Figure 4a shows the average of the accuracy measures for the kNN algorithm after increasingly injecting false packets. When there is no injected false packet, all of the devices have 91% F1 score, then it linearly decreases with the increase of false packets. For example, injecting false data equivalent to 10% of packets exchanged during the observation time resulted in a decrease by 13%. For 90% false traffic addition, the accuracy of device state detection declined by about 57%. This shows that traffic injection can be efficiently used for hiding the state of devices from the adversary.

Impact of False Data Injection on Device State Classification. We injected the falsified data into the training data and computed the accuracy metrics in terms of F1 Score, Precision, and Recall. We injected 10% falsified data and continued injecting until 90% of the dataset contained false data. As can be seen in Figure 4b, the F1 Score plunges dramatically when injecting 90% false data and reaches 15%. This is due to the fact that randomly falsified features deteriorate traffic patterns used for classifying the devices' states. Also here, we can see that by injecting increasing amounts of fabricated traffic, the adversary can effectively be prevented from making inferences about the types of device events occurring.

In real-life smart home environment, it is possible to introduce false/spoofed data packets (e.g., advertising, status/action request) by customizing the installed apps [11] using the open source environments such as Samsung SmartThings. The installed apps can generate specific data packets at a specified interval without altering the real device states to trick the attacker from detecting device states. In our implementation, we only analyzed the effectiveness of injecting spoofed network traffic into the real traffic to hide the device states. An improved defense mechanism could use a more complex strategy to hide also the location at home as well as the source and destination of packets to avoid from device state detection and classification attacks. We leave this as an open problem to be explored in our future works.

6 DISCUSSION

ISP as an adversary: Note that so far, our adversary model included only local adversary, where the adversary is within the range of radio frequency. An extension to this adversary model can be a remote adversary that can monitor outgoing network traffic of the smart home. A concrete example of such an adversary is an ISP. Compared to the local adversary model considered in this paper, an ISP-like adversary has both advantages and disadvantages. It does not have to be within a range and it can see the source and destination IPs of the packets, which a local adversary can not see if the WPA encryption is enabled. However, it can only collect the outgoing network traffic, not the internal two-way (upstream and downstream) network traffic as all the traffic is merged by the gateway (i.e., access point). Figure 5 shows the complete topology of the network from device to cloud.

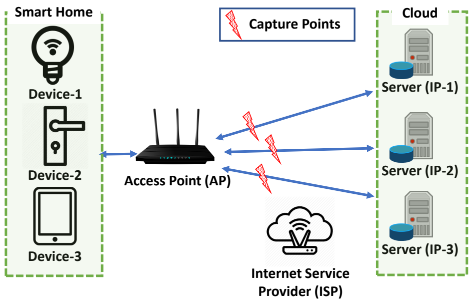


Figure 5: Remote adversary model (e.g., a malicious ISP).

As can be seen in Figure 5, an ISP will only see the router's (i.e., gateway/access point) MAC address. Therefore, it can not use the MAC addresses of the smart home devices for the device identification. However, it can still try to use IPs in order to identify the devices and infer activities. Though there are number of challenges that attacker needs to solve in order to able use IP as a device identifier. First of all, if Network Address Translation (NAT) is deployed by the AP[1], the ISP can not find out the topology of the smart home and the number of devices. Even though NAT is not enabled, ZigBee and BLE devices have never been assigned an IP as they communicate with the AP through a hub, where only the hub they are connected to gets an IP. Moreover, devices do not communicate with only one server. Instead, sometimes multiple devices use one server (i.e., destination IP) as in the Samsung ST Hub, or sometimes one device can use multiple servers [5, 14]. Therefore, even though the ISP-like attacker has some advantages (i.e., seeing IPs) over the local adversary, there are additional challenges that it needs to solve to get the same attack working. We leave this kind of adversary out of scope for now and will be studied in a future work.

Multi-user vs. single user: Smart home devices support multiple authorized users where more than one user can control and change the settings of smart devices. Additionally, multiple users can perform different activities within the smart environment at a time. This can create some false positive and false negative cases in user activity inference using our proposed method. Nonetheless, an attacker can still infer the device type and devices states from the network traffic. Additionally, the attacker can also infer the presence of multiple users and the specific point of ongoing activities in multi-user smart home environment using the network traffic. Compared to a multi-user scenario, a single user smart home environment is more vulnerable to our proposed threat as it is easier to infer a single on-going user activity in the smart home.

Local vs. remote control: To improve the user control and convenience over smart devices, smart homes offer remote access control in addition to traditional local access. Our proposed threat model can guess both local and remote access from location feature of the captured network traffic. This is a serious threat to user privacy as

[1]Assuming IPv4 is still in use.

attackers can detect when a user is changing the state of a specific device remotely and perform malicious activities. For example, an attacker can infer when a user is accessing the smart lock remotely, which may result in physical access to the home environment.

Smart device diversity: Smart devices have no common network protocols. Indeed, some of them such as WiFi, ZigBee, and BLE are more popular than others. This makes it harder to sniff all the devices that the smart home user is using. In addition to the diversity of network protocols, smart home devices come with different computational resources, hardware types, capabilities, exchanged data format etc. All of these differences in smart devices make it very challenging to build a generic solution as well as an attack. However, with our automated multi-stage privacy-attack, we showed the feasibility of the attack with the most popular network protocols, which covers the most of the commercial devices.

Generalizability of the attack: As we noted in the assumptions, the attacker we considered in this paper does not need exactly the same devices to train its attack model, but it needs exact brand and device type to get the results presented in this paper as we use the $< brand, device - type >$ pair to uniquely identify devices. In other words, we assume at the end of the device identification stage of our attack, the attacker knows $< brand, device - type >$ pair. However, this assumption weakens the attack model. An attacker who can infer the device type and does not need the same device with the same brand would be more realistic. In order to remove this assumption, the same device type with different brands should be used to train the models and to attack (i.e., testing). It would be interesting to train and test the attack models on the same device type with different brands, or the same brand with different device types. Moreover, it would be also interesting to test the affects of model numbers, device configurations, or firmware updates etc.

7 RELATED WORK

Identification using the encrypted network traffic . The metadata (e.g. MAC, traffic rate) of encrypted network traffic triggers possible threats including unintentional disclosure of the content or user. There is an extensive literature in the identification of the content from the encrypted network traffic. For example, web page identification [45], web user identification [26], protocol identification [49] are some of the research on the identification using the encrypted traffic. Not only identification attacks, but also the countermeasures have been studied in several studies [10, 19].

Fingerprinting Methods. In all the aforementioned studies, either statistical techniques [48] or machine learning methods [13, 33] were used to infer different sensitive information about the user and the context. Even ML has been used for the task of identification such as user, device, or website identification, in none of these studies, the attacks are timing-based as we have in our work.

IoT Fingerprinting. So far, in all the aforementioned studies the results showed that the used methods are efficient and the threat is real, but the threat was limited to the web and online privacy of the user. Now with the emergence of IoT, it has been extended to every part of our daily lives and, with this, threats and countermeasures have also evolved [1, 6, 41]. The number of studies on the IoT fingerprinting through the network traffic has been increasing every day. Many studies have investigated the device type identification problem, where it has been sometimes proposed for both attacking [8, 18, 28, 35, 36] and improving the security of smart home platforms [9, 31, 32]. Moreover, some other works [3–5, 14, 24, 34, 47] worked on the device activity (event) inference problem, where the phrases device activity inference and user activity inference sometimes have been used interchangeably. In our work, we refer to the device activity (event) as the activity inferred from only one device. Even though sometimes the device activity and user activity would be the same thing (e.g., "coffee maker is ON" is the same as "the user is making coffee"), sometimes information from multiple devices is needed to infer one user activity correctly (e.g., see Figure 3). We differentiate those two types of activities and provide a more generalized activity types in the fourth stage of our attack by modeling the user activities using HMM in Section 4.6.

Difference from existing work. Our work differs from the aforementioned studies in several ways: First, we are proposing a comprehensive method of end-to-end attack to infer the on-going user activities in a cascaded manner, where the previous studies have focused on only one stage of the attack. Note that putting all the different attack mechanisms and executing them successfully is a non-trivial task. Second, we are proposing the use of HMM for user activity modeling, where the device activities from multiple devices have been used to infer user activities. Last but not least, for the analysis of our attack, we performed experiments using the devices with WiFi, ZigBee, and BLE, where most of the previous studies have focused only on one of those wireless protocols.

8 CONCLUSION

In this paper, we explored how encrypted traffic from a smart home environment can be used to infer sensitive information about smart devices and sensors. Specifically, we introduced a novel multi-stage privacy attack, which an attacker can exploit to automatically detect and identify particular types of devices, their actions, states, and related user activities by passively monitoring the traffic of smart home devices and sensors. Our evaluation on an extensive list of off-the-shelf smart home devices, sensors, and real users showed that an attacker can achieve very high accuracy (above %90) in all the attack types. As opposed to to earlier straightforward activity identification approaches, the novel multi-stage privacy attack can perform detection and identification automatically, is device-type and protocol-agnostic, and does not require extensive background knowledge or specifications of analyzed protocols. Finally, we propose a new yet effective mitigation mechanism to hide the real activities of the users. The effectiveness of the multi-stage privacy attack raises serious privacy concerns for any smart environment equipped with smart devices and sensors including personal homes, residences, hotel rooms, offices of corporations or government agencies.

ACKNOWLEDGMENT

This work is partially supported by the US National Science Foundation (Awards: NSF-CAREER-CNS-1453647, NSF-1663051, NSF-1718116), Cyber Florida's Capacity Building Program, the EU LOCARD Project (agreement H2020-SU-SEC-2018-2019-2020- 8327351), the EU COLLABS Project (ICT-2019-2 GA 871518), the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - SFB 1119 - 236615297, and Cisco Systems under the IoT Crowdsense project, gift # 2019-197831 (3696). The views expressed are those of the authors only, not of the funding agencies.

REFERENCES

[1] A. Acar, H. Aksu, A. S. Uluagac, and K. Akkaya. 2020. A Usable and Robust Continuous Authentication Framework using Wearables. *IEEE Transactions on Mobile Computing* (2020), 1–1. https://doi.org/10.1109/TMC.2020.2974941

[2] IoT Analytics. 2017. State of the Smart Home Market. https://iot-analytics.com/wp/wp-content/uploads/2017/12/StateofSmartHomeMarket2017-vf.pdf.

[3] Noah Apthorpe, Danny Yuxing Huang, Dillon Reisman, Arvind Narayanan, and Nick Feamster. 2019. Keeping the Smart Home Private with Smart(er) IoT Traffic Shaping. In *Proceedings on Privacy Enhancing Technologies.* 128–148.

[4] Noah Apthorpe, Dillon Reisman, and Nick Feamster. 2017. A smart home is no castle: Privacy vulnerabilities of encrypted IoT traffic. *arXiv preprint arXiv:1705.06805* (2017).

[5] Noah Apthorpe, Dillon Reisman, Srikanth Sundaresan, Arvind Narayanan, and Nick Feamster. 2017. Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic. *arXiv preprint arXiv:1708.05044* (2017).

[6] Leonardo Babun, Amit Kumar Sikder, Abbas Acar, and A Selcuk Uluagac. 2018. Iotdots: A digital forensics framework for smart environments. *arXiv preprint arXiv:1809.00745* (2018).

[7] Bruhadeshwar Bezawada, Maalvika Bachani, Jordan Peterson, Hossein Shirazi, Indrakshi Ray, and Indrajit Ray. 2018. IoTSense: Behavioral Fingerprinting of IoT Devices. *arXiv preprint arXiv:1804.03852* (2018).

[8] Trevor Bihl, Michael Temple, and Kenneth Bauer. 2017. An Optimization Framework for Generalized Relevance Learning Vector Quantization with Application to Z-Wave Device Fingerprinting. In *Proceedings of the 50th Hawaii International Conference on System Sciences.*

[9] Simon Birnbach, Simon Eberz, and Ivan Martinovic. 2019. Peeves: Physical Event Verification in Smart Homes *(CCS '19).* ACM, New York, NY, USA, 1455–1467. https://doi.org/10.1145/3319535.3354254

[10] Xiang Cai, Xin Cheng Zhang, Brijesh Joshi, and Rob Johnson. 2012. Touching from a distance: Website fingerprinting attacks and defenses. In *Proceedings of the 2012 ACM conference on Computer and communications security.* ACM, 605–616.

[11] Z. Berkay Celik, Leonardo Babun, Amit K. Sikder, Hidayet Aksu, Gang Tan, Patrick McDaniel, and A. Selcuk Uluagac. 2018. Sensitive Information Tracking in Commodity IoT. In *USENIX Security Symposium.* Baltimore, MD.

[12] Maximilian Christ, Nils Braun, and Julius Neuffer. 2018. tsfresh. https://github.com/blue-yonder/tsfresh.

[13] M. Conti, L. V. Mancini, R. Spolaor, and N. V. Verde. 2016. Analyzing Android Encrypted Network Traffic to Identify User Actions. *IEEE Transactions on Information Forensics and Security* 11, 1 (Jan 2016), 114–125. https://doi.org/10.1109/TIFS.2015.2478741

[14] Bogdan Copos, Karl Levitt, Matt Bishop, and Jeff Rowe. 2016. Is Anybody Home? Inferring Activity From Smart Home Network Traffic. In *Security and Privacy Workshops (SPW), 2016 IEEE.* IEEE, 245–251.

[15] Asish Kumar Dalai and Sanjay Kumar Jena. 2017. WDTF: A Technique for Wireless Device Type Fingerprinting. *Wireless Personal Communications* 97, 2 (2017), 1911–1928.

[16] Aveek K Das, Parth H Pathak, Chen-Nee Chuah, and Prasant Mohapatra. 2016. Uncovering privacy leakage in ble network traffic of wearable fitness trackers. In *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications.* ACM, 99–104.

[17] Christian de Looper. 2017. The 12 best smart home devices you need to live like the Jetsons. http://www.businessinsider.com/best-smart-home.

[18] Shuaike Dong, Zhou Li, Di Tang, Jiongyi Chen, Menghan Sun, and Kehuan Zhang. 2019. Your Smart Home Can't Keep a Secret: Towards Automated Fingerprinting of IoT Traffic with Neural Networks. *arXiv preprint arXiv:1909.00104* (2019).

[19] Kevin P Dyer, Scott E Coull, Thomas Ristenpart, and Thomas Shrimpton. 2012. Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail. In *Security and Privacy (SP), 2012 IEEE Symposium on.* IEEE, 332–346.

[20] Kassem Fawaz and et al. 2016. Protecting Privacy of BLE Device Users.. In *USENIX Security Symposium.*

[21] David Formby and et al. 2016. Who's in Control of Your Control System? Device Fingerprinting for Cyber-Physical Systems.. In *NDSS.*

[22] Jun Han, Albert Jin Chung, Manal Kumar Sinha, Madhumitha Harishankar, Shijia Pan, Hae Young Noh, Pei Zhang, and Patrick Tague. 2018. Do you feel what I hear? Enabling autonomous IoT device pairing using different sensor types. In *Do You Feel What I Hear? Enabling Autonomous IoT Device Pairing using Different Sensor Types.* IEEE, 0.

[23] Mordor Intelligence. 2018. IoT Sensor Market Size - Segmented by Type (Pressure Sensor, Temperature Sensor, Proximity Sensor), End-user Industry (Healthcare, Automotive, Consumer Electronics), and Region - Growth, Trends, and Forecast (2018 - 2023). https://www.mordorintelligence.com/industry-reports/iot-sensor-market.

[24] Pierre-Marie Junges, Jérôme François, and Olivier Festor. 2019. Passive Inference of User Actions through IoT Gateway Encrypted Traffic Analysis. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM).* IEEE, 7–12.

[25] Huaxin Li, Zheyu Xu, Haojin Zhu, Di Ma, Shuai Li, and Kai Xing. 2016. Demographics inference through Wi-Fi network traffic analysis. In *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on.* IEEE, 1–9.

[26] Marc Liberatore and Brian Neil Levine. 2006. Inferring the source of encrypted HTTP connections. In *Proceedings of the 13th ACM conference on Computer and communications security.* ACM, 255–263.

[27] Yair Meidan, Michael Bohadana, Asaf Shabtai, Martin Ochoa, Nils Ole Tippenhauer, Juan Davis Guarnizo, and Yuval Elovici. 2017. Detection of Unauthorized IoT Devices Using Machine Learning Techniques. *arXiv preprint arXiv:1709.04647* (2017).

[28] Markus Miettinen, Samuel Marchal, Ibbad Hafeez, N Asokan, Ahmad-Reza Sadeghi, and Sasu Tarkoma. 2017. IoT Sentinel: Automated device-type identification for security enforcement in IoT. In *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on.* IEEE, 2177–2184.

[29] AKM Iqtidar Newaz, Amit Kumar Sikder, Mohammad Ashiqur Rahman, and A Selcuk Uluagac. 2019. Healthguard: A machine learning-based security framework for smart healthcare systems. In *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS).* IEEE, 389–396.

[30] Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Minh Hoang Dang, N Asokan, and Ahmad-Reza Sadeghi. 2018. D\" IoT: A Crowdsourced Self-learning Approach for Detecting Compromised IoT Devices. *arXiv preprint arXiv:1804.07474* (2018).

[31] TJ OConnor, William Enck, and Bradley Reaves. 2019. Blinded and confused: uncovering systemic flaws in device telemetry for smart-home internet of things. In *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks.* ACM, 140–150.

[32] TJ OConnor, Reham Mohamed, Markus Miettinen, William Enck, Bradley Reaves, and Ahmad-Reza Sadeghi. 2019. 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.* ACM, 128–138.

[33] S. V. Radhakrishnan, A. S. Uluagac, and R. Beyah. 2015. GTID: A Technique for Physical Device and Device Type Fingerprinting. *IEEE Transactions on Dependable and Secure Computing* 12, 5 (2015), 519–532.

[34] Jingjing Ren, Daniel J Dubois, David Choffnes, Anna Maria Mandalari, Roman Kolcun, and Hamed Haddadi. 2019. Information exposure from consumer iot devices: A multidimensional, network-informed measurement approach. In *Proceedings of the Internet Measurement Conference.* ACM, 267–279.

[35] Ola Salman, Imad H Elhajj, Ali Chehab, and Ayman Kayssi. 2019. A machine learning based framework for IoT device identification and abnormal traffic detection. *Transactions on Emerging Telecommunications Technologies* (2019), e3743.

[36] Mustafizur R Shahid, Gregory Blanc, Zonghua Zhang, and Hervé Debar. 2018. IoT devices recognition through network traffic analysis. In *2018 IEEE International Conference on Big Data (Big Data).* IEEE, 5187–5192.

[37] Amit Kumar Sikder, Abbas Acar, Hidayet Aksu, A Selcuk Uluagac, Kemal Akkaya, and Mauro Conti. 2018. IoT-enabled smart lighting systems for smart cities. In *Computing and Communication Workshop and Conference (CCWC), 2018 IEEE 8th Annual.* IEEE, 639–645.

[38] Amit Kumar Sikder, Hidayet Aksu, and A Selcuk Uluagac. 2017. 6thsense: A context-aware sensor-based attack detector for smart devices. In *USENIX Security.*

[39] Amit Kumar Sikder, Hidayet Aksu, and A Selcuk Uluagac. 2019. A context-aware framework for detecting sensor-based threats on smart devices. *IEEE Transactions on Mobile Computing* (2019).

[40] Amit Kumar Sikder, Leonardo Babun, Hidayet Aksu, and A Selcuk Uluagac. 2019. Aegis: a context-aware security framework for smart home systems. In *Proceedings of the 35th Annual Computer Security Applications Conference.* 28–41.

[41] Amit Kumar Sikder, Leonardo Babun, Z Berkay Celik, Abbas Acar, Hidayet Aksu, Patrick McDaniel, Engin Kirda, and A Selcuk Uluagac. 2019. Multi-User Multi-Device-Aware Access Control System for Smart Home. *arXiv preprint arXiv:1911.10186* (2019).

[42] Amit Kumar Sikder, Giuseppe Petracca, Hidayet Aksu, Trent Jaeger, and A Selcuk Uluagac. 2018. A Survey on Sensor-based Threats to Internet-of-Things (IoT) Devices and Applications. *arXiv preprint arXiv:1802.02041* (2018).

[43] Vijay Srinivasan, John Stankovic, and Kamin Whitehouse. 2008. Protecting your daily in-home activity information from a wireless snooping attack. In *Proceedings of the 10th international conference on Ubiquitous computing.* ACM, 202–211.

[44] Tim Stöber, Mario Frank, Jens Schmitt, and Ivan Martinovic. 2013. Who do you sync you are?: smartphone fingerprinting via application behaviour. In *Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks.* ACM, 7–12.

[45] Qixiang Sun, Daniel R Simon, Yi-Min Wang, Wilf Russell, Venkata N Padmanabhan, and Lili Qiu. 2002. Statistical identification of encrypted web browsing traffic. In *Security and Privacy, 2002. Proceedings. 2002 IEEE Symposium on.* IEEE, 19–30.

[46] Thomas SÃÿderholm. 2017. EU GDPR: Privacy for connected medical devices. https://blog.nordicsemi.com/getconnected/eu-gdpr-privacy-for-

connected-medical-devices.

[47] Rahmadi Trimananda, Janus Varmarken, Athina Markopoulou, and Brian Demsky. 2019. PingPong: Packet-Level Signatures for Smart Home Device Events. *arXiv preprint arXiv:1907.11797* (2019).

[48] Petr Velan, Milan Čermák, Pavel Čeleda, and Martin Drašar. 2015. A survey of methods for encrypted traffic classification and analysis. *International Journal of Network Management* 25, 5 (2015), 355–374.

[49] Charles V Wright, Fabian Monrose, and Gerald M Masson. 2006. On inferring application protocol behaviors in encrypted network traffic. *Journal of Machine Learning Research* 7, Dec (2006), 2745–2769.

A PERFORMANCE METRICS

To evaluate our proposed novel attack, we used seven different performance metrics: True Positive Rate (TPR), False Negative Rate (FNR), True Negative Rate (TNR), False Positive Rate (FPR), Precision, Accuracy, and F1-score. These can be calculated using following equations:

$$TPR(Recall) = \frac{TP}{TP + FN}, \tag{3}$$

$$FNR = \frac{FN}{TP + FN}, \tag{4}$$

$$TNR = \frac{TN}{TN + FP}, \tag{5}$$

$$FPR = \frac{FP}{TN + FP}, \tag{6}$$

$$Precision = TP/(TP + FP), \tag{7}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{8}$$

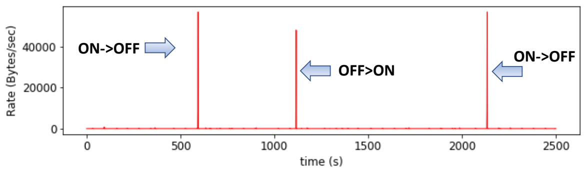
$$F1 - score = \frac{2 * TP * TN}{TP + TN}, \tag{9}$$

where $TP = True\ Positive, FP = False\ Positive, TN = True\ Negative$ and $FN = False\ Negative$.

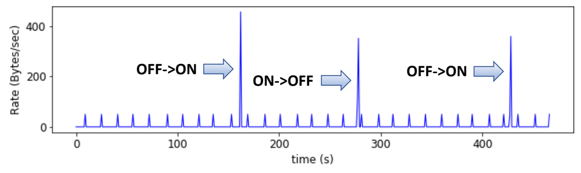
B CASE STUDIES

In this section, we show the feasibility and possibility of privacy leaks from encrypted network traffic of smart home devices. We show that an attacker who can sniff the network traffic of the devices can easily infer some simple information without using any advanced techniques. We consider one device for each protocol: Wemo Insight Switch (WiFi), Samsung ST Outlet (ZigBee), and August Smart Lock (BLE). We analyze the raw network traffic of each device and see if it is really possible to extract information from the network traffic, specifically from data rate.

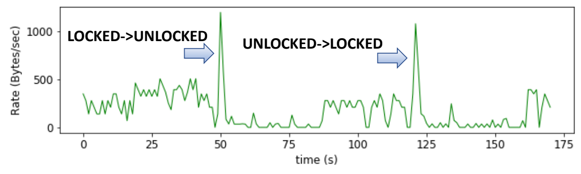
B.0.1 Wemo Insight Switch (WiFi). Wemo Insight Switch is a Wifi-enabled device and used to monitor and control other appliances (e.g., smart light) from a smartphone. It has only two capabilities: ON and OFF. Figure 6a shows the data rate of the sample traffic collected from Wemo Insight Switch, where we illustrated a number of actions of the user to change the state of the device. As can be seen from the figure, the data rate shows a significant increase when the device state is changing. Therefore, the data rate clearly reveals the device state changes. In the first peak, the device's state is changed by the user, i.e., the device is turned on and in the second peak, the user turned off the device and so on.



(a) Wemo Insight Switch (WiFi)



(b) Samsung SmartThings Outlet (ZigBee)



(c) August Smart Lock (BLE)

Figure 6: The traffic rates of (a) Wemo Insight Switch, (b) Samsung ST outlet, and (c) August Smart Lock. Here, a number of actions are illustrated, with many signals easily discerned by the naked eye. For instance, when the lock is turned on, the significant amount of packets are transmitted and received, which creates a peak in the traffic rate for a certain duration.

B.0.2 Samsung ST Outlet (ZigBee). Samsung SmartThings (ST) Outlet uses ZigBee protocol to communicate with Samsung ST Hub. It can also act as a repeater and repeats the broadcast packet of Hub for the smart devices, which is not in the range of Hub. This increase the range of Hub. Other than repeating Hub's broadcasting packets, it has only two capabilities: ON and OFF. The traffic rate of a sample network capture of Samsung ST Outlet is plotted in Figure 6b. In the given sample network traffic, the device's activity has been changed by the user three times, which clearly corresponds to the three large peaks. On the other hand, small peaks correspond to the repeating of the broadcast packets of the hub, which is periodic with 15 seconds.

B.0.3 August Smart Lock (BLE). The August Smart Lock communicates with the user's smartphone via BLE. In addition to locking and unlocking from the app on the smartphone, the owner (main user) can also give access to guest users through the web servers. The user can also enable the auto-unlock, where the lock is unlocked when the user is in range. However, the lock itself does not have the remote control capability. For remote access, it needs other accessories (e.g., WiFi bridge). Here, we only consider the BLE communication between the lock and smartphone. Figure 6c shows the plot of the sample packet capture of August Smart Lock. As in the previous case studies, the transition between the device's actions can be clearly identified by the attacker. The small increase in the traffic rate in the first part of the capture is because of the advertising packets.