FIAT: Frictionless Authentication of IoT Traffic

Yunming Xiao*
Northwestern University
yunming.xiao@u.northwestern.edu

Matteo Varvello
Nokia Bell Labs
matteo.varvello@nokia.com

ABSTRACT

Home IoT (Internet of Things) deployments are vulnerable to local adversaries, compromising a LAN, and remote adversaries, compromising either the accounts associated with IoT devices or third-party devices like mobile phones used to control the IoT. There is, however, a fundamental difference between an attacker and a legitimate IoT user: the physical interaction with the device (e.g., via a mobile app) used to operate the IoT. Such physical interactions can be used to build *frictionless* authentications. However, their integration with IoT requires each vendor to independently adopt them, which is both complex and expensive. We instead design and build *FIAT*, the first *third-party* mechanism to automatically authorize IoT traffic by learning recurring traffic and validating human actions behind unpredictable traffic. FIAT does not require modification of the IoT devices or apps, as it operates passively on network traffic. Our evaluation shows that FIAT achieves high accuracy with minimal impact on the user experience.

CCS CONCEPTS

• **Security and privacy → Network security**; **Authentication**; • **Networks → Home networks**.

ACM Reference Format:
Yunming Xiao and Matteo Varvello. 2022. FIAT: Frictionless Authentication of IoT Traffic. In *The 18th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '22), December 6–9, 2022, Roma, Italy*. ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3555050.3569126

1 INTRODUCTION

The average US household currently hosts more than 10 Internet of Things (IoT) devices [6]. Many research papers [25, 33, 58, 63] and blogs [1, 3, 7] have demonstrated critical security concerns of the IoT, often due to lack of best practices like partial usage of HTTP, or old ciphers. Even when best security practices are implemented, the IoT is still vulnerable to many attacks. For example, intruders can penetrate the home WiFi and directly control some IoT devices [75]. They can compromise the account associated with an IoT device, mostly relying on username and password, or of third party services like IFTTT [14]. They can also compromise the devices where IoT apps run, *i.e.,* mostly mobile phones [2].

The above security concerns could be mitigated via two-factor authentication (2FA), as commonly done for online banking. With 2FA, the user is often required to validate her identity via, for instance, an SMS received on a mobile phone. Unfortunately, requiring a user to constantly validate her interactions with IoT devices is cumbersome, and unlikely to be accepted by users – which is why it is not adopted.

Recently, a few solutions were proposed to apply *frictionless* authentication, which does not interrupt the user experience, for IoT via biometric recognition [37, 47, 68]. These solutions are *first party*, *i.e.,* they need to be developed and supported by each IoT vendor. While simple to implement, these solutions have two main drawbacks which have, so far, hindered their adoption. First, they require a modification of existing IoT devices and applications. Second, they require each IoT vendor to independently implement them.

The (ambitious) goal of this work is to build a *third-party* frictionless authentication mechanism for IoT devices, which can thus be deployed today across most existing IoT devices without any vendor support. Our rationale is that IoT traffic is highly *predictable* – packets with a constant size are sent to a single destination at a constant pace – as mostly caused by software, e.g., frequently reporting temperature readings from a smart thermostat. Less frequently, this traffic originates from "routines" set by the user, e.g., "turn on the heat at 6pm", or by a user via "manual" input, e.g., increase the temperature from the thermostat app. Predictable traffic can be learned and automatically authorized. Unpredictable traffic, when legitimate, is associated with some physical interaction between the user and a controlling device. We thus plan to automatically validate unpredictable traffic leveraging sensor data (e.g., accelerometer and gyroscope) from the mobile phone used to control an IoT device.

Our first contribution is a quantification of the predictability of IoT traffic. We do so by analyzing public datasets and

*This work was done during the internship at Nokia Bell Labs.

CoNEXT '22, December 6–9, 2022, Roma, Italy

ACM ISBN 978-1-4503-9508-3/22/12…$15.00
https://doi.org/10.1145/3555050.3569126

via a 10-devices testbed we have deployed at two US locations (Illinois and New Jersey). The analysis of public datasets shows that 80-90% of the IoT traffic (from hundred of devices) is indeed predictable. Figure 1(a) shows an intuition of the reason behind this predictability: 8 highly predictable flows generated by the Bose SoundTouch 10 as observed in the YourThings dataset [23]. We confirm this result in our testbed, showing even higher predictability (about 98% on average) thanks to precise labeling and full traffic access.

Our second contribution is a demonstration that it is possible to accurately and quickly identify manual traffic, *i.e.,* traffic that is associated with a user action, by leveraging machine learning tools. We extract unpredictable packets from the trace collected at our testbed and group them into labeled *events*. We select features based on the first (up to) 5 packets for each unpredictable event, and apply various machine learning classification algorithms, among which the highest balanced accuracy across the 10 IoT devices is reached by the Nearest Centroid Classifier [51] (0.93) and Bernoulli Naive Bayes [52] (0.91).

Our third contribution is the design and implementation of *FIAT*, the first third-party frictionless authentication mechanism for IoT traffic. FIAT aims at improving the security of legacy IoT devices with no user input for authentication. We explain FIAT via an example. The user launches the Nest app on her phone, and lowers the temperature which causes the AC to turn on. FIAT's *app* (on the phone) detects that the Nest app was launched and starts collecting gyroscope and accelerometer data. Next, it leverages a pre-shared key – securely stored in the device's trusted execution environment – to sign this data and quickly transfer it to FIAT's *IoT proxy* (using QUIC's zero-RTT [4]), a secure device previously installed in the home. Meanwhile, the actual IoT command is sent to Nest/Google cloud and then down to the target device, where it is intercepted by the IoT proxy. This traffic is authorized granted that the IoT proxy has verified that *a human was interacting with the Nest app on a pre-authorized device.*

We implemented a prototype of FIAT as an Android service and a Rasberry Pi acting as the IoT proxy for the devices in our testbed. We then run several experiments involving automated commands, a real user, and both experiments from LAN and mobile. The evaluation shows that FIAT's traffic authentication is always faster than actual IoT traffic, both when the user is on LAN (by >74%) or a mobile network (by >50%). Further, FIAT accurately classifies IoT traffic which produces very low false positives and false negatives – zero for half of the devices and at most 6% for the other half. High accuracy and low latency translate to no noticeable impact on the user experience.

We organize the paper as the following. In Section 2, we introduce our method for determining whether the traffic is predictable and present our measurement of predictability on public datasets. In Section 3, we introduce the event definition and further conduct finer-grained predictability analysis on different events from our own testbed. Next, we investigate how to distinguish the unpredictable manual events in Section 4. With all the above results and insights, we introduce our design of FIAT in Section 5 and evaluate the system in Section 6. Limitations and future work are discussed in Section 7. Last, we summarize related work in Section 8 and conclude in Section 9.

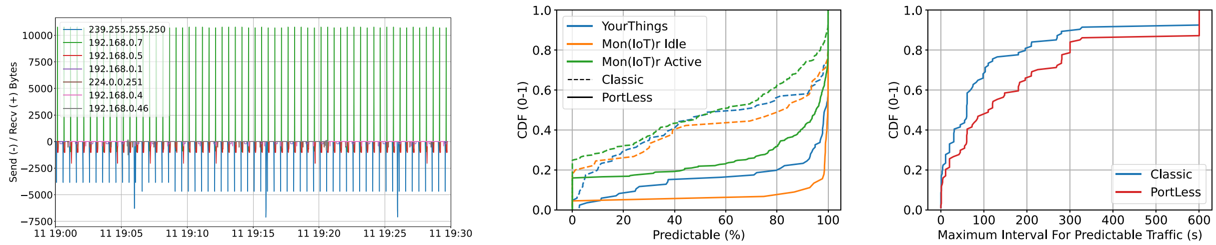
2 IS IOT TRAFFIC PREDICTABLE?

This section investigates the fundamental assumption that motivates the design of FIAT, *i.e.,* that IoT traffic is highly predictable. Related works have shown that IoT traffic has unique patterns which allow accurate passive IoT device identification, even leading to potential privacy concerns. However, to the best of our knowledge, no previous study has yet quantified how predictable IoT traffic is, *i.e.,* how often such per-device patterns tend to repeat over time. Further, it is unclear what is the impact of the traffic type, *i.e.,* distinguishing between *control* traffic, needed by the device to operate, *automated* traffic, *i.e.,* traffic triggered by routines like from IFTTT [14], and *manual* traffic, *i.e.,* human-triggered traffic caused by a user interacting with an IoT device through its companion app. We are interested in answering these questions to motivate and drive the design of FIAT. In the remainder of this section, we first introduce a heuristic to determine the predictability of IoT traffic, which we then investigate leveraging large public datasets.

2.1 Methodology

We say that packets are *predictable* if they have the same size and are sent to or received from the same IP address or domain name at a constant rate. To investigate the predictability of IoT traffic, we proceed as follows. For each packet, we record arrival timestamp, size, source and destination IPs, transport protocol (TCP/UDP), and source and destination ports. We then store each packet in a bucket identified by the tuple above minus the arrival timestamp. We then compute the inter-arrival times between packets from the same bucket considering the last two received packets. If the computed inter-arrival time matches any previously computed inter-arrival times for this bucket, then all packets associated with this inter-arrival time (previous or future) are considered *predictable*.

We have observed that, over time, some IoT devices regularly communicate with the same destination (domain name) while using different port numbers. Thus, in addition to the above 6-tuple $< ip\_src, ip\_dst, port\_src, port\_dst, proto, size >$ ("Classic"), we introduce a "PortLess" 4-tuple, which abandons $< port\_src, port\_dst >$, and further replace ip_dst with



(a) (b) (c)

Figure 1: Evaluation of the predictability of IoT traffic. (a) TCP/UDP flows for Bose SoundTouch 10 over 30 minutes. (b) CDFs of the percentage of predictable traffic in YourThings and Mon(IoT)r datasets. Classic vs PortLess flow definition. (c) Maximum intervals of predictable flows in the YourThings dataset.

its associated domain name[1]. Figure 1(a) shows a visual example of the predictability for 8 flows generated by the Bose SoundTouch as observed in the YourThings dataset [23, 25].

2.2 Results

We explore two large and publicly available datasets: YourThings [23, 25] and Mon(IoT)r [64]. The YourThings dataset includes the network traffic collected from 65 IoT devices during 10 days (106 GB). Mon(IoT)r dataset includes the network traffic collected from 104 IoT devices and 16 controller devices, e.g., an Android phone used to turn up/down the volume of a smart speaker via its official app. The Mon(IoT)r dataset is divided into *idle* (4.1 GB) and *active* (8.8 GB) traffic. Idle traffic refers to testing scenarios with no human-initiated action at any controller. Active traffic relates to the IoT traffic triggered by an operation performed at its companion app. Active traffic is also collected shortly before and after an action is executed; unfortunately, the traffic generated by the device where the action is executed was not collected.

Figure 1 shows Cumulative Distribution Functions (CDFs) of the percentage of predictable traffic across devices, distinguishing between the "Classic" and "PortLess" approaches described above. We start by focusing on the YourThings dataset. Figure 1(b) shows that more than 80% of the traffic for 80% of the devices is predictable, assuming the PortLess approach. It is quite possible that part of this unpredictable traffic is *manual*; unfortunately, this dataset does not contain traffic labels which allow further analysis.

Next, we focus on the Mon(IoT)r dataset, distinguishing between idle and active traffic (Figure 1(b)). Given routines were not explored in this dataset, we assume that idle traffic only contains control traffic, and active traffic contains a combination of control and manual traffic. Figure 1(b) shows high predictability of idle (control) traffic, e.g., up to 90% of the traffic for 90% of the devices considering PortLess approach. In contrast, when there are active actions invoked the (manual) IoT traffic predictability is reduced.

Finally, Figure 1(c) shows the maximum intervals of predictable packets for all devices in the YourThings dataset. The Mon(IoT)r dataset cannot be used for this analysis since it does not provide continuous traffic collection. The figure shows that 80-90% of the predictable traffic occurs regularly within 5 minutes, and the maximum interval is 10 minutes. It follows that, within 2x of the maximum interval, *i.e.,* 20 minutes, of traffic capturing, all predictable traffic (*i.e.,* 80% for this dataset) can be potentially identified.

IoT Inspector [40] also provides a large dataset collected from real home IoT deployments. However, it does not provide packet-level granularity but only coarse-grained five-second information aggregation – which will reduce the predictability to a great extent. For instance, one unpredictable packet will change the sum of packet sizes over a 5-second window and make that window fully unpredictable. In contrast, if we have the packet-level granularity, the unpredictability is limited to that unpredictable packet only. Still, we perform a similar analysis on the sample data of IoT Inspector by iterating through the 5-second information aggregations rather than the packets. The results show that half of the devices have a predictability greater than 85% given PortLess definition.

3 IOT TRAFFIC ANALYSIS FROM OUR TESTBED

The previous analysis has shown potential for IoT traffic predictability, except for manual traffic – originated by human interactions with the devices – or possibly automated traffic. However, both datasets have some limitations for the analysis we are performing. 1) The YourThings dataset does

[1]We obtain the domain name either from DNS requests – when available in the trace – or via a reverse DNS lookup. As we send the reverse DNS lookup requests to the same recursive resolver in Illinois, the same IP will correspond to the same domain name. Hence, it has at least the same accuracy as directly using the IP addresses. Nevertheless, such reverse lookups are not as accurate as DNS requests in the trace because of the presence of domain aliases. This might also be one of the reasons for the higher predictability of the traffic in our own dataset than that in the public dataset (see Section 3).

Table 1: Testbed and experiments description.

Location	Model	Brand	Quantity	Description	Command (Automation)
New Jersey, Japan (VPN), Germany (VPN)	Echo Dot 4	Amazon	1	Smart speaker	Play Music (Reminders) (IFTTT - alert)
	Home Mini	Google	1	Smart speaker	Play music Tell phone battery Turn on/off SP10 (Reminders) (IFTTT - alerts)
	Wyze-Cam	Wyze	3	Camera	Watch monitor Take a photo Configure settings (Camera turn on) (Upload a short video)
	SP10	Teckin	3	Smart plug	Turn on/off (Turn on/off)
Illinois	Home	Google	1	Smart speaker	Play music (Reminders)
	Nest-E	Google	2	Thermostat	(Turn on/off) (Change temperature)
	Echo Dot 3	Amazon	1	Smart speaker	Play Music (Reminders) (IFTTT - alert)
	E4 Mop Robot	Roborock	1	Robot Vacuum	Clean room Check status (Clean room)
	Blink Camera	Amazon	1	Camera	Watch monitor Configure settings (Camera turn on) (Upload a short video)
	WP3	Gosund	2	Smart plug	Turn on/off (Turn on/off)

not offer labeled traffic, e.g., it is impossible to distinguish control versus manual traffic. 2) The Mon(IoT)r dataset does not provide continuous traffic collection, often missing the beginning of a connection, *i.e.,* the TCP/TLS handshake. 3) No dataset explores IoT routines. Motivated by these observations, we proceed to build our own testbed, experiments, and data collection.

3.1 Testbed

Table 1 describes the 10 IoT devices which compose our testbed, and which experiments were performed on each device. The device selection is based on what we had at our disposal. In Appendix B, we discuss the popularity of such devices as well as whether they are present in public datasets. The testbed is deployed at two locations: New Jersey and Illinois. The New Jersey location is a controlled environment including also an Android device (Samsung Galaxy S10) and a Rasberry Pi. The Android device is used to execute the set of actions described in Table 1 (last column) via the companion app of each IoT device, e.g., the Alexa app for the Echo Dot. The Rasberry Pi acts as a 2.4 GHz WiFi access point where IoT devices and phone connect to. This allows to monitor the traffic produced by each app and IoT device. Further, the Rasberry Pi is used to run a VPN client (provided by ProtonVPN [19]) emulating different network locations (Germany and Japan) for all the IoT devices and mobile phone.

The Illinois location is instead a real household with a single user who was provided with an Android app to log ground truth data of when she was interacting with a given app. Note that this app cannot report *which* action was performed – and thus Table 1 reports possible actions executed – but only when and for how long the user had the IoT companion app open. For traffic collection, we have deployed a Rasberry Pi which uses ARP spoofing [76] to intercept and collect all IoT traffic. This approach was preferred upon using a controlled WiFi which would cause the inconvenience to reset the WiFi at all IoT devices. We collect traces for 15 days, which contain about 20 interactions per IoT device, on average. The most frequent used devices are are two smart plugs (40 interactions), whereas the least used is the E4 Mop Robot (8 interactions).

We set up routines with each IoT device's companion app or IFTTT as described in Table 1 within the parentheses. At the NJ location, we perform humans-like interactions by randomly selecting which IoT device to interact with, for how long, and how long to wait before the next human-like interaction. These humans-like interactions with the IoT apps were realized by the Rasberry Pi using the Android Debugging Bridge (ADB) [9]. We collect traces for two weeks for all the devices and VPN configurations.

3.2 Predictability

We identify (un)predictable packets for each device using the method described in Section 2. We further use time timestamps from the routine setup and the logs of manual operations to label traffic as automated and manual – and then control for all other traffic. We group unpredictable packets into unpredictable *events* as follows: given a series of unpredictable packets that arrive at $T_1, T_2, \ldots, T_N$, we first create an unpredictable event E_1 that includes the first packet. Then, we iterate through the unpredictable packets; if $T_2 - T_1 < 5$ seconds – this threshold was chosen empirically and has very limited impact on the results –, then E_1 is extended to include the second packet. The procedure continues until $T_N - T_{N-1} > 5$ seconds. Then we consider that E_1 ends at $(N-1)^{th}$ packet, and we create a new unpredictable event E_2 that includes the N^{th} packet. The procedure continues until there are no more unpredictable packets.

Figure 2 shows the traffic predictability per IoT device and category (control, automated, manual). The figure shows that the control traffic is overall highly predictable: around 98% for all devices, given the PortLess definition, confirming what is reported in Section 2. The Nest thermostat is an outlier, with only 90.7% of predictable control traffic. Further investigation reveals lots of unpredictable "events" (as per above definition) happening every hour but with slightly different intervals (ranging from a few to ten seconds). Without access to its code, we cannot conclude what is responsible

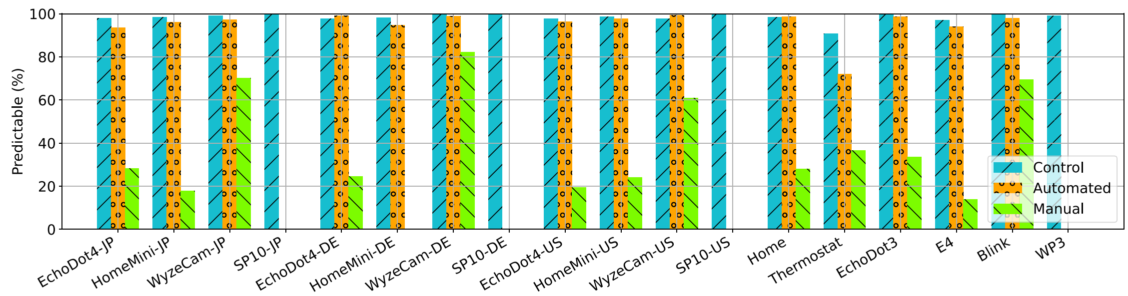


Figure 2: Predictability of control, automated, and manual traffic in our testbed using the PortLess flow definition.

for these events. However, it has to be noted that: 1) the Nest thermostat is equipped with a motion sensor which allows it to turn on its screen when a person is passing by; 2) the Nest thermostat is capable to turn off, for instance, when no mobile phone is detected in the same LAN. These behaviors are highly dependent on user behavior, and can thus generate unpredictable "control" traffic.

With respect to automated traffic, the figure shows its predictability is overall lower than control traffic. Still, most of the devices have a predictability of around 90%. This happens because such automation is still controlled by software and thus, within an automation, its traffic is largely repetitive. The unpredictable automated events contain a minimum of 2 packets (SP10 and WP3) and at most 30 packets (Google Home). Because there are no predictable packets between those two packets for SP10 and WP3, the figure shows a predictability of 0. One could attempt to predict those 2 packets with a daily frequency, for instance, but we deliberately avoided this to: 1) present a worst case, 2) avoid the complexity of having to deal with dynamic routines (e.g., depending on dynamic behaviors like "at sunset"). Finally, the predictability of manual traffic is, overall, the worst. This depends on the way such commands are realized, with potential random user behaviors. However, the manual traffic for IoT cameras (WyzeCam and Blink) has higher predictability (60-65%) than other devices. This is because video streaming tends to generate packets at a constant rate, which are predictable given the constant inter-arrival times (as per our methodology in Section 2.1). Similar to automated traffic, manual events for SP10 and WP3 only contain two unpredictable packets, and thus have a predictability of 0.

3.3 Communication Models

We have manually investigated the IoT traffic traces collected to further understand each device *communication model* or how traffic is exchanged between phone, IoT device, and cloud, in presence of unpredictable automated and manual events. We find one communication model for unpredictable automated events shared by all devices, and three for unpredictable manual events. We here summarize the main findings which are useful to drive the design of FIAT.

Traffic Direction – IoT devices keep at least one persistent connection to the cloud, through which they are "notified" of automated or manual commands. These commands trigger the creation of new connections between the IoT device and the phone, either direct when the mobile phone is in the same LAN as the IoT device or via a relay server otherwise. This implies that potential un-authorized traffic can be found both *incoming* and *outgoing* the IoT device.

Command Duration – To prevent un-authorized IoT commands from completing, it is crucial to identify unauthenticated traffic quickly, *i.e.,* before the first N packets, where N is the smallest number of packets that allows the command to complete its function. Our traffic investigation has revealed that each IoT device is characterized by a different N. For instance, a simple IoT device like the smart plug SP10 just needs one 235 B packet to turn on/off, whereas a complex device like Google Home sends/receives up to hundreds of packets just when the user opens the app. We experimented with the minimum number of packets required by each IoT device to correctly execute manual commands. To do so, we start from $N = 1$ and increase N until the command is correctly executed. We find that N ranges from 1 (SP10 and WP3) to 41 (WyzeCam).

Location – We further investigated whether the communication models change per location (US, Germany, and Japan). We find that all devices still follow the same communication models, but some might interact with their cloud provider using not only different destination IPs, expected due to geolocation, but also different domain names. For example, Google Home will talk to *google.co.jp* when it is located in Japan rather than *google.com* when in the US.

4 MANUAL TRAFFIC CLASSIFICATION

The previous sections have shown that most of control and automated IoT traffic is predictable using the simple heuristic described in Section 2. However, to achieve FIAT's authentication goal with low false positives, we need to further distinguish manual traffic from unpredictable control and automated traffic. For simple IoT devices, like SP10, WP3,

Table 2: Model selection.

Model	Mean Balanced Accuracy
Nearest Centroid Classifier	**0.931**
Bernoulli Naive Bayes	**0.906**
Neural Network	0.786
Gaussian Naive Bayes	0.779
Decision Tree	0.745
AdaBoost Classifier	0.739
Support Vector Classifier	0.713
Random Forest	0.706
K-Nearest Neighbors	0.621

and Nest-E, we can construct rules to identify manual traffic by visually inspecting their traffic; for example, the size of the notification packets (267 and 235 Bytes) is a distinctive feature for manual traffic directed to both IoT devices. Therefore, we exclude these devices from the analysis in this section. However, manual inspection is impractical for more complex devices like Google Home; not to mention that, even for the same device, the rule/pattern can change even for different versions or locations.

In this section, we investigate whether machine learning is an appropriate tool to classify the unpredictable manual IoT traffic. We first explore different machine learning models. Next, we focus on the most effective models, and detail the classification results. Last, we investigate the "transferability" of such ML models among devices at different locations.

4.1 Model And Feature Selection

Based on related works [28, 29, 45, 61, 79] and our observations so far in the paper, we select 66 features for event classification among which: packet's direction (*i.e.,* whether sent or received by the device), remote IP address, protocol, TCP flags, source and destination ports, TLS version, packet length, and packets inter-arrival times. The features also include statistics such as mean of packet sizes and inter-arrival times between unpredictable packets.

Following related literature [28, 29, 45, 61, 79] on traffic classification, we have tested numerous ML models as listed in Table 2, which shows the best results among all the hyperparameters that we have experimented with each ML model. We exclude SP10, WP3, and Nest-E in the experiments because a simple rule on the packet size is enough. The balanced accuracy assigns the same weight to all traffic: control, automated, and manual. We use balanced accuracy to reduce the impact of different numbers of unpredictable control/automated/manual events in our dataset.

We pre-process all the data by scaling all the features to unit variance before training and testing with ML models. For Nearest Centroid Classifier (NCC) and k-Nearest Neighbors (kNN), we have tested different distance metrics, including Euclidean distance, Manhattan distance, and Chebyshev distance. For NCC, we find that Chebyshev distance performs the best. For kNN, Euclidean distance has the best performance. Further, we have tested different values of k for kNN ranging from 3 to 15, of which it has the best performance when $k = 5$. For neural networks, we adopt 128 as the hidden layer size. We have explored the number of hidden layers from 1 to 10. The neural network with 8 hidden layers has the best performance in our dataset. For the decision tree, we have tested different maximum depths from 2 to 12. We find that a decision tree with a maximum depth to be 3 has the best performance.

Table 3: Unpredictable manual event classification.

Device	Nearest Centroid Classifier Precision	Recall	F1 S.	Bernoulli Naive Bayes Precision	Recall	F1 S.
EchoDot4-US	0.74	0.80	0.77	0.83	0.93	0.88
EchoDot4-JP	0.74	0.83	0.78	0.85	0.89	0.87
EchoDot4-DE	0.75	0.88	0.81	0.94	0.94	0.94
HomeMini-US	0.81	1.00	0.90	0.84	1.00	0.91
HomeMini-JP	0.86	0.96	0.91	0.80	1.00	0.89
HomeMini-DE	0.97	1.00	0.98	0.89	0.99	0.94
WyzeCam-US	0.83	0.89	0.86	0.87	0.88	0.87
WyzeCam-JP	0.98	1.00	0.99	0.90	0.94	0.92
WyzeCam-DE	0.98	0.94	0.96	0.98	1.00	0.99
Home	0.82	0.71	0.76	0.73	0.81	0.77
EchoDot3	0.89	0.92	0.90	0.93	0.96	0.94
E4	0.76	1.00	0.86	0.86	0.75	0.80
Blink	0.91	1.00	0.95	0.91	1.00	0.95

4.2 Results

In this subsection, we focus on the two most effective ML algorithms (NCC and BernoulliNB) for a detailed evaluation. Table 3 shows how these two ML algorithms perform on the classification tasks for different IoT devices, focusing only on unpredictable manual event classification due to space limit. The results refer to the mean from five-fold cross-validation. When available, results from multiple (VPN) locations of the devices are reported.

Table 3 lists precision, recall, and F1 score (harmonic average of precision and recall) of the unpredictable manual event classification for all IoT devices when using NCC and BernoulliNB. Both algorithms perform well for EchoDot3, Blink, WyzeCam, and HomeMini (F1 scores > 0.9) but relatively poor on Google Home (F1 scores < 0.8). For the EchoDot4 , BernoulliBN has relatively higher F1 scores than BB, *i.e.,* 0.9 versus 0.8. With devices under VPN, we find that Germany and Japan have slightly better results than the US. From the precision we can infer that in the worst case (EchoDot4 with NCC), up to 25% of the unpredictable events that are classified to be manual are actually unpredictable control/automated events. Given that unpredictable manual events account for 2% of the total events, this 25% only accounts for 0.5% of all events.

Table 4: Features ranked by permutation importance score for WyzeCam-DE.

Feature Name	Permutation Importance
pkt1-proto	0.0737
pkt1-direction	0.0076
pkt3-tls	0.0059
pkt3-tcp-flags	0.0042
pkt1-tls	0.0025
......	
pkt1-dst-ip1	0.0000
pkt1-dst-ip2	0.0000
pkt1-dst-ip3	0.0000
pkt1-dst-ip4	0.0000
pkt2-dst-ip1	0.0000
......	

4.3 Knowledge Transfer

We explore how the ML models work at a high level, with the goal to comment on their "transferability", *i.e.,* if a model learned for a device at location X can be used by a device at location Y. Note that the transferability only applies to the ML models that classify the unpredictable events, whereas the heuristic of identifying predictable traffic (Section 2) is instead thought per device and location and thus cannot be transferred – this is because of its dependency on IP/domain, which is very much sensitive to geolocation.

We adopt permutation feature importance to understand which feature(s) play an important role in the classification. Specifically, for each input feature we randomly shuffle its values across all the data points (events). In this case, the F1 score for the model is expected to decrease if an input feature is important. The permutation importance is then defined to be the score difference. We iterate 50 times for each feature to get reliable results.

The ranking of feature permutation importance may be different for every device and ML model. Here we demonstrate an example of training WyzeCam-DE trace with BernoulliNB, as it has shown a nearly perfect result (F1 score of 0.99). Table 4 shows the feature permutation importance of both top and bottom ranked features. We find that the transport protocol, packet direction, and TLS version play the most important roles in successfully classifying the traffic type. Conversely, the IP addresses do not play any role in the classification (permutation importance equal to zero).

We proceed to verify the transferability of the ML models. Table 5 shows the F1 scores when we train the models with data from one location and test with data from another location. The table shows that the F1 scores are very high for all the devices for both NCC and BernoulliNB models. In fact, the F1 scores are higher than the cross validation within the same dataset at the same location (Table 3). There are two reasons for the higher scores: 1) larger training set compared to cross validation – which cannot train with all the data; 2) the ranges of some features, e.g., IP addresses, are changed. For example, when considering a single location, the model might mistakenly learn that the chance of an unpredictable manual event increases a bit when the destination IP starts with "172". When we expand the dataset considering additional locations, now the IP destination never starts with "172", correcting the previous learning error. Either way, the high F1 scores verify that the IP addresses are not important features and the knowledge of how to classify unpredictable events can be transferred.

Table 5: F1 score of transfer.

Device	Transfer	Nearest Centroid Classifier F1 Score	Bernoulli NB F1 Score
EchoDot4	US-JP	0.94	0.97
	US-DE	0.93	0.98
	JP-DE	0.94	0.97
HomeMini	US-JP	0.98	0.98
	US-DE	0.98	0.98
	JP-DE	0.99	0.98
WyzeCam	US-JP	0.94	0.97
	US-DE	0.94	0.97
	JP-DE	0.97	1.00

5 FIAT DESIGN AND IMPLEMENTATION

This Section designs and implements FIAT, a frictionless authentication mechanism for IoT traffic. FIAT aims at improving the security of IoT devices without disrupting their functioning, *i.e.,* with no impact on their current traffic or requiring annoying user action validation. FIAT automatically learns *control* and *automated* traffic, thanks to its demonstrated predictability, and leverages humanness verification to handle the unpredictable *manual* traffic. In the following, we first describe our threat model, and then proceed with the description of FIAT and its components.

5.1 Threat Model

We assume a computationally bounded attacker who can compromise any IoT account of the user, either of a specific app like SmartThings [20] or of centralized services like IFTTT [14], or Google Home [13]. We further assume an attacker who can control the home network, e.g., by breaking WiFi security, and can inject, drop, reroute, and modify (unencrypted) packets, but cannot break cryptographic primitives [32]. We also assume the attacker can compromise any of the devices associated with FIAT, for example by installing spyware applications [60] which can read sensor data, detect active applications, etc. However, we assume the attacker has no access to the device's OS level, e.g., (s)he cannot fake sensor data such as gyroscope and/or accelerometer. Finally, we assume attackers cannot hack into Trusted Execution Environments (TEEs).

5.2 Overview

Figure 3 visualizes the main components of FIAT. On the left end-side, the figure shows an Android device running FIAT's client-side component, a user-space application that

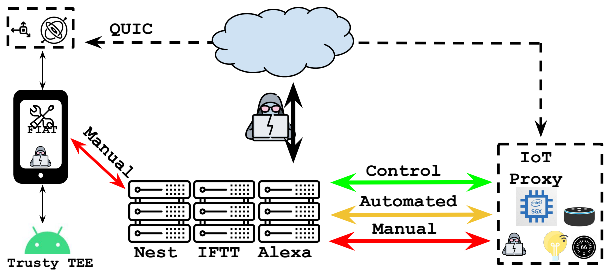


Figure 3: Graphical view of FIAT's architecture.

leverages the device's TEE as a hardware-backed secure keystore [12]. In the following, we simply refer to it as FIAT's app. The center of the figure shows the IoT traffic, distinguishing between control, automated, and manual. The figure further shows some *new* traffic (carried over QUIC) originated by FIAT's app; this traffic carries a proof of human interaction linked with an IoT app. The right end-side of the figure shows instead a typical home network, with a smart bulb and FIAT's server-side component. This is a secure IoT proxy, e.g., implemented over SGX [53], which intercepts IoT traffic and also receives the traffic carrying the human input validation. The figure further shows potential attackers as per our threat model: *remote* attackers who have access to the user's IoT account and/or the user-space of the device, and *local* attackers who have penetrated the home WiFi.

5.3 Client-Side App

FIAT's app monitors user interaction with IoT apps, and quickly and securely informs the IoT proxy of this interaction. This allows the IoT proxy to verify the validity of eventual *manual* IoT traffic, *i.e.,* that the traffic requesting to turn on a smart light is associated with the user physically interacting with the mobile app of the smart light.

We have implemented FIAT's app as an Android service. The app monitors the current IoT app in use via the accessibility service permission [10], manually enabled by the user. Each time an IoT app is in use, e.g., to turn on a light, the app starts collecting the device's sensor data. We collect sensors data (accelerometer and gyroscope) using `SensorManager` and `SensorEventListener` at highest frequency (250 samples per second). The sensor data, along with information on which IoT app is in use, is encrypted with a key obtained by the TEE's keystore (using Jetpack security [15]), and sent to the IoT proxy. As detailed later, this key is agreed offline between FIAT's app and IoT proxy at *pairing*. The human verification data is sent to the IoT proxy via a fast channel so that the proxy is informed of the human activity before that the corresponding manual traffic (triggered by a user interacting with the IoT app) is intercepted. We now describe each of these operations in more detail.

Human Detection – Whenever a (human) user interacts with the mobile's display, the force applied during the touch action generates motion. This motion is captured by embedded sensors like accelerometer and gyroscope. Lack of changes in the values reported by the sensors is instead indicative of a potential attacker who has either compromised the IoT account, or the device (e.g., simulating user touches). This observation has been used to build frictionless CAPTCHA alternatives like Invisible CAPPCHA [36] and zkSENSE [62], a privacy-preserving solution which does not leak sensor data at the expense of longer computation time. In FIAT, we are not concerned about leaking sensor data given this information is only shared with the IoT proxy which is owned by the user. Accordingly, FIAT's app reports raw sensor data – or more precisely features extracted as per the ML model discussed below – to the IoT proxy which runs the humanness validation algorithm.

Fast and Secure Channel – QUIC is a UDP-based, encrypted transport protocol recently standardized by IETF [4]. QUIC is the perfect tool to build a fast and secure channel between FIAT's client and server-side components for the following reasons. First, QUIC (0-RTT or 1-RTT) allows to save, at least, one RTT required when setting up a TCP connection. Second, QUIC encrypts both data and metadata, *i.e.,* transport information, leaving little to none information available to an intermediary. QUIC 0-RTT is however vulnerable to replay attacks [34], where an adversary can reuse a previously sent package (without modification). This attack is a concern for FIAT; however, given only few devices are authorized within a household, it is feasible for the IoT proxy to keep a state of all previously held connections, which would prevent a replay attack [74]. We integrate QUIC support in FIAT's app using the Cronet library for Android [11].

5.4 Server-Side IoT Proxy

Figure 4 illustrates the position and the whole procedure of the FIAT's proxy. At a high level, FIAT's proxy intercepts all the packets destined to IoT devices in the home network. It then follows that the packets will need to go through the access control of FIAT's proxy. The access control consists of two heuristics including predictability by bucket match as in Section 2.1 and event grouping as in Section 3.2, and two machine learning algorithms including unpredictable manual event classifier as in Section 4 and humanness verification. We will introduce the humanness verification in later this subsection. The input of all the access control components is or is derived from the incoming packets from the IoT vendors' cloud servers; the only exception is the humanness verification, whose input is the FIAT authentication message from another channel – directly from the mobile device of the user. To enable the FIAT authentication message, a pairing between the user's mobile device and the FIAT's proxy is needed beforehand. The output of the access control is

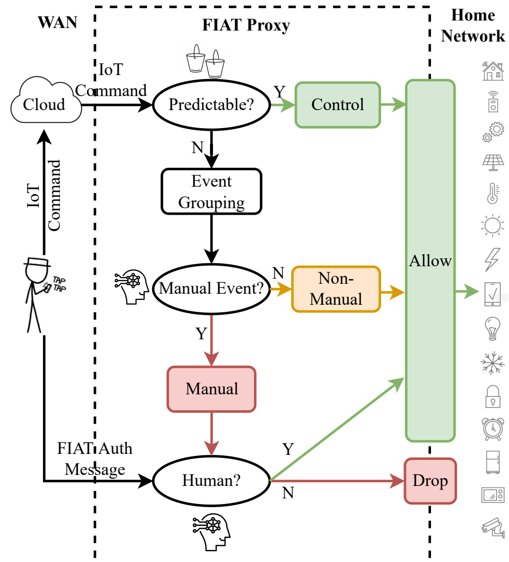


Figure 4: Graphical view of FIAT proxy's procedure.

either to allow the packet to proceed to the IoT devices in the home network, or to drop the packets. We now describe each task in detail.

Traffic Intercept – Similar to IoT inspector [40], or products like Circle parental control [17], we intercept IoT traffic via ARP spoofing [76]. This allows for quick deployment without the need to integrate with an actual home gateway. We set up iptables for all the forwarded traffic to an NFQUEUE [21], which delays the packet forwarding and submits the whole packets to a userspace Linux application, which runs our traffic analysis. We then wait for the application to decide whether to proceed with the forwarding or to drop the packet, which is then executed in kernel space.

Rules Creation – Figure 1(c) shows that up to 20 minutes are needed to start correctly predicting control IoT traffic from our devices. During this *bootstrapping* time, all traffic is allowed and FIAT's IoT proxy populates several access control rules [8]. These rules describe flows using the "PortLess" definition given its superior performance. This process is per device. We do not attempt to transfer the learned rules of predictable traffic because of the various IoT devices, versions, and potential different behaviors at different locations.

Access Control – IoT traffic is matched against the rules created by the heuristic introduced in Section 2.1. In case of a rule *hit*, the packet is considered predictable and is allowed. In case of a rule *miss*, the packet is considered unpredictable. We use the same mechanism as introduced in Section 3.2 to group the unpredictable packets into unpredictable events. The first N packets (if there are any) of every unpredictable event are allowed. N depends on the associated device, as empirically estimated in the previous section. We then feed the features extracted from the first N packets of the unpredictable event to the classifier of the corresponding device (via simple rules or the BernoulliNB model). The classifier returns the type of unpredictable event: control, automated, manual.

If the unpredictable event is classified as non-manual, all packets associated with this event are allowed to pass. Otherwise, the IoT proxy only allows packets from this event if FIAT's app has already verified the human activity responsible for this traffic. If the latter condition does not verify, then event packets are dropped and the user is notified of a potential security breach. Further, if this condition repeats multiple times under a short period of time, *i.e.,* a potential brute-force attack, the device is disconnected until the activity is manually verified by the user.

Human Input Validation – In [62], the authors have investigated the performance of four ML-based classifiers (SVM, decision tree, random forest, and neural network with ReLU kernel) as mechanisms to identify human interaction with a mobile device. Overall, the classifiers achieve similar performance (0.95 recall), and gyroscope and accelerometer data were identified as the most accurate features. We thus opted for the same technique for FIAT's humanness identification mechanism. We use the same ML model which performs the best, *i.e.,* 9-layer decision tree, and trained with the same data as in the previous study [62], where the inputs are 48 features extracted from the gyroscope and accelerometer.

Pairing – We assume that FIAT's IoT proxy and app are paired locally. For example, by scanning a QR code on the proxy or listening to a sound emitted when the proxy is connected to the home router. At pairing, the app is authenticated at the IoT proxy and the agreed keys are safely stored in their respective TEEs: Android secure keystore and SGX. The IoT proxy rejects any traffic which does not validate: 1) from an unauthorized device, 2) from an authorized device failing the humanness validation.

6 EVALUATION

We set up FIAT proxies at both households from our testbed (see Section 3.1) and pair them with the FIAT's app on the corresponding phones. We assume a bootstrapping time of 20 minutes during which FIAT allows all traffic and learns how to distinguish predictable from unpredictable packets (see Section 4). Next, we perform several experiments to evaluate FIAT's *accuracy* and *latency.* [2]

[2]We adopt the simple rules for the access control of SP10, WP3, and Nest-E (see Section 5). For the other devices, we choose the BernoulliNB model with default parameters of sklearn package [24] as the manual event classifier given its high accuracy overall and better transferability than NCC (see Section 4), where the first $N = 5$ packets of each unpredictable event are allowed and are considered as the inputs to the BernoulliNB model.

Table 6: FIAT accuracy evaluation.

	Precision/Recall of Event Classifier (%)		Precision/Recall of Human Validation (%)		FIAT (%)		
Device	**Manual**	**Non-Manual**	**Human**	**Non-Human**	**False Positive Manual**	**False Positive Non-M.**	**False Negative**
Echo Dot 4	94.2/98.0	99.5/98.5	99.2/93.4	93.8/98.2	1.40	1.76	3.76
Home Mini	96.1/98.0	99.3/98.7			1.21	1.76	3.76
WyzeCam	100/100	100/100			0.00	0.00	0.00
SP10	100/100	100/100			0.00	0.00	0.00
Home	96.1/98.0	99.2/98.3			1.59	1.76	3.76
Nest-E	100/100	100/100			0.00	0.00	0.00
Echo Dot 3	94.3/100	100/98.6			1.31	0.00	0.00
E4	92.3/96.0	97.0/95.5			3.76	1.72	5.72
Blink	100/100	100/100			0.00	0.00	0.00
WP3	100/100	100/100			0.00	0.00	0.00

Table 7: FIAT latency evaluation; LAN/Mobile

	Wyze	**Socket**	**EchoDot**	**Home Mini**
IoT operation	*Get video*	*Turn on/off*	*Play the radio*	*Play music*
Time to first packet	1130/1182ms	692/891ms	622/792ms	1396/1970ms
Time to human validation (0-RTT)	154/235ms	141/352ms	161/394ms	152/223ms
App Detection	85/65ms	61/75ms	87/68ms	79/66ms
Sensor sampling	235/246ms	251/249ms	247/258ms	259/243ms
Secure storage access	45.5/52.2ms	55.6/48.5ms	50.4/48.7ms	49.4/53.1ms
QUIC (1-RTT)	27.5/270ms	27.6/602ms	27.4/1044ms	26.1/233ms
QUIC (0-RTT)	21.8/115ms	23.2/226ms	21.2/275ms	21.6/102ms
ML-based human validation	2.05/2.65ms	2.06/2.84ms	2.62/2.47ms	2.00/2.09ms

Accuracy Analysis – We automate the manual operations from Table 1 using ADB, assuming the same routines are also set. We use automation to generate a large set of manual activities – 500 operations, 50 per device – while also evaluating the accuracy of the (non-)human verification. Next, we ask the IL user to naturally interact with her IoT devices over a week with the Android phone equipped with the FIAT app. This experiment generates about 100 IoT interactions, and consequent human validations at the IoT proxy.

Table 6 (left end-side) shows, for each device, the recall of unpredictable manual and non-manual (control/automated) events. These results refer to 50 unpredictable manual events per device along with 60-180 unpredictable non-manual (control+automated) events. The table shows, overall, very high recall for both manual (>0.96 for all devices) and non-manual (>0.98 for all devices) classification. Further, the event classifier performs perfectly (100% precision and recall) with WyzeCam, SP10, Nest-E, Blink, and WP3. The E4 MopRobot has the worst recall, overall; this is due to the relatively small training dataset because of the low-frequency usage of the mop robot in our household testbed (IL). The middle part of Table 6 shows also high recall of human (0.934) and non-human (0.982) verification.

Next, we evaluate FIAT's false positives and false negatives. False positives happen in two scenarios. First, when FIAT incorrectly blocks control traffic or routines (unpredictable control/automated events) because they are misclassified – while the lack of human activity is correctly detected (otherwise the traffic is allowed). Second, when FIAT incorrectly blocks manual traffic (unpredictable manual events) – despite being correctly classified – because the human activity is incorrectly not validated. False negatives happen when an unpredictable manual event is incorrectly classified as control/automated, or it is correctly classified but the human behavior is incorrectly validated. False positives may block a legit IoT function to execute whereas false negatives can lead to a successful attack if synchronized with an attacker. For a more formal explanation, please refer to Appendix A.

Table 6 shows that FIAT's false positive/negative rate is quite low, indeed zero for WyzeCam, SP10, Nest-E, Blink, and WP3. With respect to false positives, they are zero for 5 out of 10 devices and a few percentages for the others. Our visual investigation and reporting from the user in IL did not report any issue in the functioning of the devices. This happens for two reasons: 1) control traffic is mostly unnoticeable to the user; given IoT devices are not real-time, a delay of a few seconds (for example in reporting a temperature reading) is hard to notice; 2) these devices are programmed to "retry" few times (even routines and manual events) and would thus eventually succeed (if legit). We will later comment on a similar behavior with respect to FIAT latency analysis.

The most vulnerable device is the E4 MopRobot, with a false negative rate, *i.e.,* chance of a successful attack, of 5.72%. Note that 4% out of 5.72% comes from misclassification of unpredictable manual events, which is due to the user performing some "complex" interactions with the MopRobot app which was not covered in the training dataset – due to little amount of data reported for this device (Section 3.1).

Latency Analysis – We consider two scenarios of FIAT usage: LAN and mobile network in the home proximity. For each scenario, we repeat five times the activity described in Table 1 for the NJ devices – which are under our control. For the *mobile* experiments, we add a Mint mobile [18] SIM to the Android device, connect the Rasberry Pi to a power bank and drive for roughly one hour, *i.e.,* until all tests are completed, within 15 miles radius from the home network.

Table 7 shows the breakdown of the latency analysis per device, operation, and scenario (each cell refers to the average latency measured on LAN/mobile, respectively). We assume a worst-case scenario for FIAT, where the *time to first packet* is computed from when the IoT command is sent (via ADB in this test) and not when the app is launched. This is to avoid biased due, for instance, a slower phone. Table 7 shows that, using QUIC 0-RTT, FIAT *always authenticates manual traffic faster than it is received.* QUIC 0-RTT not only reduces the network latency, but it also offers faster execution time, both in Android and the Rasberry Pi. Note that, when calculating the "time to human validation" we have ignored the time for sensor sampling for two reasons. First, in the case of QUIC 1-RTT sensor sampling can happen in parallel with the connection setup. In the case of QUIC 0-RTT, we assume the FIAT app can keep a lazy buffer of sensor data, *i.e.,* subscribe to sensor events in low frequency and increase the frequency when an IoT app is detected in the foreground – which requires about 60-80ms.

Finally, we investigate how *slow* can FIAT afford to be before breaking IoT functionalities. We add synthetic latency to FIAT humanness validation and quantify *when* the functioning of an IoT device gets impaired. We empirically verified that all IoT devices can tolerate two seconds extra delay. This is because the additional delay is managed by TCP – used by all devices in our testbed – which adapts to the sudden RTT change via timeout and retransmissions.

7 DISCUSSION

Limitations – FIAT suffers from potential false positives and false negatives, which come from the inaccuracy of machine learning models applied in both unpredictable manual event classification and humanness verification. A larger training dataset would substantially decrease the false negative rate. Regardless, no ML model is perfect and false positives and false negatives are to be expected when applying ML to security [28, 77]. However, an attacker still has little chance of exploiting such false positive rate for two reasons: 1) it requires reverse engineering the ML model, 2) it requires brute forcing which is protected by adding some friction to the system (see Section 5.4).

With respect to false negatives, they might degrade the user experience if they block a manual operation from completion. One potential solution for false negatives is to allow some authentication friction, which also degrades the user experience. Nevertheless, an optimal level of friction might exist maximizing the user experience. We leave the exploration as future work.

Potential Attack – An attacker can install a spyware on one of the trusted devices which can know when the user controls the device, and listen to which application is in the foreground. Let's say this attacker is attempting to open a garage door remotely. The attacker can synchronize the attack when the user launches the garage door app, for example, to check if the door is locked. This attack succeeds, *i.e.,* the garage door will open, because the attacker is piggybacking on actual human activity on an authorized device. However, the attacker is restricted to the time when the user is interacting with a targeted IoT device. The attacker cannot piggyback an IoT app running in the background. It is noteworthy that two-factor authentication (2FA) cannot handle such powerful attackers. Even worse, 2FA without human verification (e.g., via an SMS which can be read by a spyware) does not require the attacker to sync with user activity.

Complex Scenarios – In some scenarios, there are interactions between the IoT devices in smart homes. For example, some smart lights can be controlled by Alexa. By default, the command from Alexa to the smart light will be dropped by FIAT because no humanness is verified at the same time. This can be resolved by adding a rule that allows all the unidirectional traffic from Alexa to the smart light at FIAT 's proxy. This may lead to a set of rules following a Directed Acyclic Graph (DAG) among the IoT devices in smart homes. We have envisioned this to be part of our future work. A further question might be how to make sure Alexa is not "hacked" by someone shouting from outside. As FIAT is a general solution, such special cases should be handled by the Alexa developers, e.g., by adding a "voice match" function [16].

Technology Acceptance – One may raise concern regarding the silent false negative in FIAT, *i.e.,* a successful attack may occur without noticing the users. This concern is common for biometric recognition when machine learning is used. In contrast, one argument might be that 2FA, while cumbersome, provides better technology acceptance to the users because the text message may alert the user of unapproved attempts. But in fact, 2FA is worse as it turns out to be the solution that provides a false sense of security because a powerful attacker with access to the mobile phone can easily intercept and delete the message before the user is notified [22, 31]. Instead, FIAT's decisions are made in FIAT's proxy, which also keeps logs of all the unpredictable events (regardless of whether they are manual/non-manual or authenticated/unauthenticated). To delete the records, the attacker is required to have access to the home network and access into TEEs of the proxy, where the latter hacking power is out of the scope of our threat model (see Section 5.1). Therefore, it would be not practical for an attacker, who has the same hacking power needed to exploit the 2FA, to be able to alter the FIAT's records. Reporting such logs to the users can effectively relieve the concerns and allow the users to notice the silent false negatives. While this function is not explored in this paper, they are certainly achievable by FIAT and are considered to be our future work.

Road to Production – This paper has evaluated FIAT in a test-bed with up to 10 devices. Building a larger scale test-bed is challenging, both from a logistic and cost perspective. We believe that the best way to further experiment with FIAT is to open source it, so that other research groups can test it with their own devices. It would be particularly beneficial to integrate FIAT with Mon(IoT)r [64] and IoT Inspector [40], which are already deployed large-scale IoT testbeds.

Another important angle which we did not explore in this paper is investigating better feature selection and more machine learning models. In addition to permutation importance (see Section 4), other techniques such as SHAP [65, 72] would help to verify/measure the effectiveness of each feature, and hence help the feature selection. We plan to also experiment with temporally-relevant models, e.g., LSTM [39], to handle the temporal variation in devices' behaviors [43].

For a production version of FIAT, we envision one model per IoT device and software version which is downloaded and applied automatically as FIAT identifies a new device. Device identification is not the focus of this study but solutions from the related work [43] (see "Device Identification" in Section 8) could be applied to FIAT.

8 RELATED WORK

Device Identification – The authors of [70] investigate whether simple port-based detection is effective to uniquely identify IoT devices. They find this simple technique to be effective for 14 out of 18 devices they tested. Meidan et al. [54] propose a more general approach using machine learning (ML) which achieves 99% accuracy across 9 devices. Many follow-up papers explore additional ML techniques and expand to more devices [43, 56, 59, 66, 69]. Recently, Apthorpe et al. [40] have released a crowdsourcing tool which collects labeled network traffic from IoT devices, aiming to maintain a curated dataset of IoT traffic at scale and over time.

Privacy Concerns – The above papers have demonstrated that passive IoT device identification is possible, which might lead to violation of user privacy. Ren et al. [64] further investigate whether IoT traffic exposes private and sensitive information, e.g., personal identifiable information and recordings of user activity. They find that 72 out of 81 devices they investigated expose some information to the non-first party, and all devices have at least one plaintext flow. Apthorpe et al. [27] further demonstrate privacy leakages from IoT traffic even when their traffic is encrypted, due to the uniqueness of their characteristics like DNS queries and send/receive rates. Recently, IoT traffic padding is proposed as a defense against privacy leakage from IoT traffic [26]. In addition, Manadalari et al. [50] investigate whether *all* traffic is essential to the correct functioning of the IoT. They find that 16 out of 31 devices in their testbed have at least 1 and up to 11 blockable non-essential traffic.

Security Threats – Alrawi et al. [23, 25] show that 19 out of 45 devices they tested have at least 1 and up to 6 security concerns in the traffic they send/receive, including SSL issues, susceptibility to man-in-the-middle attacks, and more. Fernandes et al. [33] quantify the risk of joining third-party automation services like IFTTT [14], and propose a platform that mitigates the risks by constraining the privileges of such services. Rahmati et al. [63] explore a similar problem and propose Tyche, a risk-based permission model that provides fine-grained risk control. To better tackle such security threats, the IETF is proposing the Manufacturer Usage Description (MUD), which formally specifies the purpose of IoT devices [5]. Hamza et al. [38] have provided an open source tool that assists IoT manufacturers in implementing the MUD.

Frictionless Authentication in IoT – Frictionless authentication does not interrupt the user experience, as required by CAPTCHA or 2FA. Instead, it continuous authenticates the user in the background without requiring any specific user action such as solving a puzzle or receiving an SMS. Biometric recognition leverages mobile sensors [36, 62, 78], keystroke [35, 71], contexts [55], trace histories [48, 49], or multi-factor combined, to achieve frictionless authentication. Researchers have integrated frictionless authentication in IoT scenarios [37, 47, 68]. These solutions are first party, *i.e.,* they assume support from IoT vendors. Our approach (FIAT) is instead a third-party solution requiring no vendor support.

Traffic Classification – Traffic classification is the basis for many network applications.Early approaches include mapping applications to transport ports [41, 67] and deep packet inspection [30, 67], before they are no longer effective since widespread use of dynamic ports and encrypted connection. Recent traffic classification attempts often leverage machine learning techniques [42, 44, 46, 57, 73] with the transport layer characteristics such as packet sizes as inputs.

9 CONCLUSION

This paper has presented FIAT, the first third-party frictionless mechanism for IoT traffic authentication. By being third-party, FIAT provides security to IoT devices without requiring each vendor to independently integrate frictionless authentications, which is both complex and expensive. FIAT is built on the idea that the majority of IoT traffic is *predictable, i.e.,* it can be learned and translated into access control rules at a proxy. Unpredictable traffic is instead due to automated events, e.g., triggered by routines from IFTTT, or manual events, which are caused by either a user physically interacting with IoT apps, e.g., to turn on a light, or by an attacker. FIAT distinguishes the unpredictable manual events from unpredictable automated events using simple rules and machine learning techniques, and then combines traffic analysis with automated human input verification to ensure IoT traffic is constantly monitored and verified. We evaluate FIAT with a deployment in a testbed consisting of 10 IoT devices spread between a controlled lab and real household. Results show that FIAT achieves all its designed goals with no false-positives and false-negatives for half of the devices and minor for the other half (a rate of 1-5%).

REFERENCES

[1] Backdooring the Frontdoor, 2016. https://av.tib.eu/media/36251.

[2] Automation script, 2019. https://github.com/NEU-SNS/intl-iot/blob/master/moniotr/auto_experiments/auto_app.sh.

[3] How to hack an IoT device, 2019. https://eandt.theiet.org/content/articles/2019/06/how-to-hack-an-iot-device/.

[4] QUIC: A UDP-Based Multiplexed and Secure Transport, 2019. https://datatracker.ietf.org/doc/html/rfc9000.

[5] RFC 8520: Manufacturer Usage Description Specification, 2019. https://datatracker.ietf.org/doc/html/rfc8520.

[6] Average number of connected devices residents have access to in U.S. households in 2020, by device, 2020. https://www.statista.com/statistics/1107206/average-number-of-connected-devices-us-house.

[7] Hacking into Internet Connected Light Bulbs, 2020. https://www.contextis.com/en/blog/hacking-into-internet-connected-light-bulbs.

[8] acl(5) — Linux manual page, 2021. https://man7.org/linux/man-pages/man5/acl.5.html.

[9] Android Debug Bridge (adb), 2021. https://developer.android.com/studio/command-line/adb.

[10] Android Developers: AccessibilityService, 2021. https://developer.android.com/reference/android/accessibilityservice/AccessibilityService.

[11] Android Developers: Perform network operations using Cronet, 2021. https://developer.android.com/guide/topics/connectivity/cronet.

[12] Data Encryption on Android with Jetpack Security, 2021. https://android-developers.googleblog.com/2020/02/data-encryption-on-android-with-jetpack.html.

[13] Google Home Mini - Smart Speaker For Any Room, 2021. https://store.google.com/us/product/google_home_mini_first_gen.

[14] IFTTT helps every thing work better together, 2021. https://ifttt.com/.

[15] JetPack: The Best WordPress Security Plugin, 2021. https://jetpack.com/features/security/.

[16] Link your voice to your devices with Voice Match - Google Assistant Help, 2021. https://support.google.com/assistant/answer/9071681?hl=en&co=GENIE.Platform%3DAndroid.

[17] Meet Circle: Parental Control and Internet Filtering App, 2021. https://meetcircle.com/.

[18] Mint Mobile, 2021. https://www.mintmobile.com/.

[19] ProtonVPN: Protect yourself online, 2021. https://protonvpn.com/.

[20] SmartThings: One simple home system, a world of possibilities, 2021. https://www.smartthings.com/.

[21] The netfilter.org "libnetfilter_log" project, 2021. https://netfilter.org/projects/libnetfilter_queue.

[22] Using SMS for two-factor authentication is dangerous, 2021. https://hackcontrol.org/blog/sms-two-factor-authentication-dangerous/.

[23] YourThings Data, 2021. https://yourthings.info/data/.

[24] sklearn.naive_bayes.BernoulliNB, 2022. https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.BernoulliNB.html.

[25] Alrawi, O., Lever, C., Antonakakis, M., and Monrose, F. Sok: Security evaluation of home-based iot deployments. In *2019 IEEE symposium on security and privacy (sp)* (2019), IEEE, pp. 1362–1380.

[26] Apthorpe, N., Huang, D. Y., Reisman, D., Narayanan, A., and Feamster, N. Keeping the smart home private with smart (er) iot traffic shaping. *Proceedings on Privacy Enhancing Technologies 2019*, 3 (2019), 128–148.

[27] Apthorpe, N., Reisman, D., and Feamster, N. A smart home is no castle: Privacy vulnerabilities of encrypted iot traffic. *arXiv preprint arXiv:1705.06805* (2017).

[28] Buczak, A. L., and Guven, E. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications surveys & tutorials 18*, 2 (2015), 1153–1176.

[29] Chao, S.-C., Lin, K. C.-J., and Chen, M.-S. Flow classification for software-defined data centers using stream mining. *IEEE Transactions on Services Computing 12*, 1 (2016), 105–116.

[30] Dewes, C., Wichmann, A., and Feldmann, A. An analysis of internet chat systems. In *Proceedings of the 3rd ACM SIGCOMM Internet Measurement Conference, IMC 2003, Miami Beach, FL, USA, October 27-29, 2003* (2003), pp. 51–64.

[31] Dmitrienko, A., Liebchen, C., Rossow, C., and Sadeghi, A. On the (in)security of mobile two-factor authentication. In *Financial Cryptography and Data Security - 18th International Conference, FC 2014, Christ Church, Barbados, March 3-7, 2014, Revised Selected Papers* (2014), N. Christin and R. Safavi-Naini, Eds., vol. 8437 of *Lecture Notes in Computer Science*, Springer, pp. 365–383.

[32] Dolev, D., and Yao, A. On the security of public key protocols. *IEEE Transactions on information theory 29*, 2 (1983), 198–208.

[33] Fernandes, E., Rahmati, A., Jung, J., and Prakash, A. Decoupled-ifttt: Constraining privilege in trigger-action platforms for the internet of things. *arXiv preprint arXiv:1707.00405* (2017).

[34] Fischlin, M., and Günther, F. Replay attacks on zero round-trip time: The case of the tls 1.3 handshake candidates. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)* (2017), IEEE, pp. 60–75.

[35] Giuffrida, C., Majdanik, K., Conti, M., and Bos, H. I sensed it was you: authenticating mobile users with sensor-enhanced keystroke dynamics. In *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment* (2014), Springer, pp. 92–111.

[36] Guerar, M., Merlo, A., Migliardi, M., and Palmieri, F. Invisible cappcha: A usable mechanism to distinguish between malware and humans on the mobile iot. *computers & security 78* (2018), 255–266.

[37] Gupta, S. *Next-generation user authentication schemes for iot applications*. PhD thesis, Ph. D. dissertation, University of Trento, Italy, 2020.

[38] Hamza, A., Ranathunga, D., Gharakheili, H. H., Roughan, M., and Sivaraman, V. Clear as mud: Generating, validating and applying iot behavioral profiles. In *Proceedings of the 2018 Workshop on IoT Security and Privacy* (2018), pp. 8–14.

[39] Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural Comput. 9*, 8 (1997), 1735–1780.

[40] Huang, D. Y., Apthorpe, N., Li, F., Acar, G., and Feamster, N. Iot inspector: Crowdsourcing labeled network traffic from smart home devices at scale. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 4*, 2 (2020), 1–21.

[41] Karagiannis, T., Broido, A., Faloutsos, M., and Claffy, K. C. Transport layer identification of P2P traffic. In *Proceedings of the 4th ACM SIGCOMM Internet Measurement Conference, IMC 2004, Taormina, Sicily, Italy, October 25-27, 2004* (2004), pp. 121–134.

[42] Kim, H., Claffy, K. C., Fomenkov, M., Barman, D., Faloutsos, M., and Lee, K. Internet traffic classification demystified: myths, caveats, and the best practices. In *Proceedings of the 2008 ACM Conference on Emerging Network Experiment and Technology, CoNEXT 2008, Madrid, Spain, December 9-12, 2008* (2008), p. 11.

[43] Kolcun, R., Popescu, D. A., Safronov, V., Yadav, P., Mandalari, A. M., Mortier, R., and Haddadi, H. Revisiting iot device identification. In *5th Network Traffic Measurement and Analysis Conference, TMA 2021, Virtual Event, September 14-15, 2021* (2021), V. Bajpai, H. Haddadi, and O. Hohlfeld, Eds., IFIP.

[44] Li, W., and Moore, A. W. A machine learning approach for efficient traffic classification. In *15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2007), October 24-26, 2007, Istanbul, Turkey* (2007), pp. 310–317.

[45] Liang, E., Zhu, H., Jin, X., and Stoica, I. Neural packet classification. In *Proceedings of the ACM Special Interest Group on Data Communication*. 2019, pp. 256–269.

[46] Liang, E., Zhu, H., Jin, X., and Stoica, I. Neural packet classification. In *Proceedings of the ACM Special Interest Group on Data Communication, SIGCOMM 2019, Beijing, China, August 19-23, 2019* (2019), pp. 256–269.

[47] Ma, Y., Wu, J., Long, C., and Lin, Y.-B. Mobidiv: A privacy-aware real-time driver identity verification on mobile phone. *IEEE Internet of Things Journal* (2021).

[48] Mahbub, U., and Chellappa, R. Path: person authentication using trace histories. In *2016 IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)* (2016), IEEE, pp. 1–8.

[49] Mahbub, U., Komulainen, J., Ferreira, D., and Chellappa, R. Continuous authentication of smartphones based on application usage. *IEEE Transactions on Biometrics, Behavior, and Identity Science 1*, 3 (2019), 165–180.

[50] Mandalari, A. M., Dubois, D. J., Kolcun, R., Paracha, M. T., Haddadi, H., and Choffnes, D. Blocking without Breaking: Identification and Mitigation of Non-Essential IoT Traffic. In *Proc. of the Privacy Enhancing Technologies Symposium (PETS)* (2021).

[51] Manning, C., Raghavan, P., and Schütze, H. Vector space classification. *Introduction to Information Retrieval* (2008), 289–317.

[52] McCallum, A., Nigam, K., et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization* (1998), vol. 752, Citeseer, pp. 41–48.

[53] McKeen, F., Alexandrovich, I., Berenzon, A., Rozas, C. V., Shafi, H., Shanbhogue, V., and Savagaonkar, U. R. Innovative instructions and software model for isolated execution. *Hasp@ isca 10*, 1 (2013).

[54] Meidan, Y., Bohadana, M., Shabtai, A., Guarnizo, J. D., Ochoa, M., Tippenhauer, N. O., and Elovici, Y. Profiliot: a machine learning approach for iot device identification based on network traffic analysis. In *Proceedings of the symposium on applied computing* (2017), pp. 506–509.

[55] Miettinen, M., Heuser, S., Kronz, W., Sadeghi, A.-R., and Asokan, N. Conxsense: automated context classification for context-aware access control. In *Proceedings of the 9th ACM symposium on Information, computer and communications security* (2014), pp. 293–304.

[56] Miettinen, M., Marchal, S., Hafeez, I., Asokan, N., Sadeghi, A.-R., and Tarkoma, S. Iot sentinel: Automated device-type identification for security enforcement in iot. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)* (2017), IEEE, pp. 2177–2184.

[57] Moore, A. W., and Zuev, D. Internet traffic classification using bayesian analysis techniques. In *Proceedings of the International Conference on Measurements and Modeling of Computer Systems, SIGMETRICS 2005, June 6-10, 2005, Banff, Alberta, Canada* (2005), pp. 50–60.

[58] Obermaier, J., and Hutle, M. Analyzing the security and privacy of cloud-based video surveillance systems. In *Proceedings of the 2nd ACM international workshop on IoT privacy, trust, and security* (2016), pp. 22–28.

[59] Ortiz, J., Crawford, C., and Le, F. Devicemien: network device behavior modeling for identifying unknown iot devices. In *Proceedings of the International Conference on Internet of Things Design and Implementation* (2019), pp. 106–117.

[60] Pierazzi, F., Mezzour, G., Han, Q., Colajanni, M., and Subrahmanian, V. A data-driven characterization of modern android spyware. *ACM Transactions on Management Information Systems (TMIS) 11*, 1 (2020), 1–38.

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

[62] Querejeta-Azurmendi, I., Papadopoulos, P., Varvello, M., Nappa, A., Zhang, J., and Livshits, B. zksense: A friction-less privacy-preserving humanattestation mechanism for mobile devices. *Proc. of the Privacy Enhancing Technologies Symposium (PETS)* (2021).

[63] Rahmati, A., Fernandes, E., Eykholt, K., and Prakash, A. Tyche: A risk-based permission model for smart homes. In *2018 IEEE Cybersecurity Development (SecDev)* (2018), IEEE, pp. 29–36.

[64] Ren, J., Dubois, D. J., Choffnes, D., Mandalari, A. M., Kolcun, R., and Haddadi, H. Information Exposure for Consumer IoT Devices: A Multidimensional, Network-Informed Measurement Approach. In *Proc. of the Internet Measurement Conference (IMC)* (2019).

[65] Ribeiro, M. T., Singh, S., and Guestrin, C. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016* (2016), B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, and R. Rastogi, Eds., ACM, pp. 1135–1144.

[66] Santos, M. R., Andrade, R. M., Gomes, D. G., and Callado, A. C. An efficient approach for device identification and traffic classification in iot ecosystems. In *2018 IEEE Symposium on Computers and Communications (ISCC)* (2018), IEEE, pp. 00304–00309.

[67] Sen, S., Spatscheck, O., and Wang, D. Accurate, scalable in-network identification of p2p traffic using application signatures. In *Proceedings of the 13th international conference on World Wide Web, WWW 2004, New York, NY, USA, May 17-20, 2004* (2004), pp. 512–521.

[68] Shila, D. M., and Srivastava, K. Castra: Seamless and unobtrusive authentication of users to diverse mobile services. *IEEE Internet of Things Journal 5*, 5 (2018), 4042–4057.

[69] Sivanathan, A., Gharakheili, H. H., Loi, F., Radford, A., Wijenayake, C., Vishwanath, A., and Sivaraman, V. Classifying iot devices in smart environments using network traffic characteristics. *IEEE Transactions on Mobile Computing 18*, 8 (2018), 1745–1759.

[70] Sivanathan, A., Gharakheili, H. H., and Sivaraman, V. Can we classify an iot device using tcp port scan? In *2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS)* (2018), IEEE, pp. 1–4.

[71] Stanciu, V.-D., Spolaor, R., Conti, M., and Giuffrida, C. On the effectiveness of sensor-enhanced keystroke dynamics against statistical attacks. In *proceedings of the sixth ACM conference on data and application security and privacy* (2016), pp. 105–112.

[72] Strumbelj, E., and Kononenko, I. Explaining prediction models and individual predictions with feature contributions. *Knowl. Inf. Syst. 41*, 3 (2014), 647–665.

[73] Sun, H., Xiao, Y., Wang, J., Wang, J., Qi, Q., Liao, J., and Liu, X. Common knowledge based and one-shot learning enabled multi-task traffic classification. *IEEE Access 7* (2019), 39485–39495.

[74] Syverson, P. A taxonomy of replay attacks [cryptographic protocols]. In *Proceedings The Computer Security Foundations Workshop VII* (1994), IEEE, pp. 187–191.

[75] Visoottiviseth, V., Akarasiriwong, P., Chaiyasart, S., and Chotivatunyu, S. Pentos: Penetration testing tool for internet of thing devices. In *TENCON 2017-2017 IEEE Region 10 Conference* (2017), IEEE, pp. 2279–2284.

[76] Whalen, S. An introduction to arp spoofing. *Node99 [Online Document]* (2001).

[77] Xiao, L., Wan, X., Lu, X., Zhang, Y., and Wu, D. Iot security techniques based on machine learning: How do iot devices use ai to enhance security? *IEEE Signal Processing Magazine 35*, 5 (2018), 41–49.

[78] Yang, Y., Sun, J., and Guo, L. Personaia: A lightweight implicit authentication system based on customized user behavior selection. *IEEE Transactions on Dependable and Secure Computing 16*, 1 (2016), 113–126.

[79] Zhang, H., Lu, G., Qassrawi, M. T., Zhang, Y., and Yu, X. Feature selection for optimizing traffic classification. *Computer Communications 35*, 12 (2012), 1457–1471.

A FORMAL ANALYSIS OF FIAT FALSE POSITIVE/NEGATIVE PROBABILITIES

Let $P\{X|Y\}$ denote the probability that Y is classified or determined to be X, given either the unpredictable event classifier or the humanness validation. For instance, $P\{non\_manual|manual\}$ denotes the probability of misclassification of unpredictable manual events to be non_manual (control/automated) ones, or $P\{non\_human|human\}$ denotes the probability of mis-validation of human behaviors to be non_human ones. Let R denote the recall; then the recalls of manual event, non_manual event, human behavior, and non_human behavior (the second value in the second to fifth row in Table 6) are R_{manual}, R_{non_manual}, R_{human}, R_{non_human}, respectively. Thus we have:

$$\begin{cases} P\{manual|manual\} & = R_{manual} \\ P\{non\_manual|manual\} & = 1 - R_{manual} \\ P\{manual|non\_manual\} & = 1 - R_{non\_manual} \\ P\{non\_manual|non\_manual\} & = R_{non\_manual} \end{cases}, \quad (1)$$

and

$$\begin{cases} P\{human|human\} & = R_{human} \\ P\{non\_human|human\} & = 1 - R_{human} \\ P\{human|non\_human\} & = 1 - R_{non\_human} \\ P\{non\_human|non\_human\} & = R_{human} \end{cases}. \quad (2)$$

In FIAT, false positives and false negatives depend on the combination of unpredictable event classifier and humanness validation. False positives happen in two scenarios. First (FP-N), when FIAT incorrectly blocks control traffic or routines (unpredictable control/automated events) because they are misclassified ($P\{manual|non\_manual\}$) – while the lack of human activity is correctly detected ($P\{non\_human|non\_human\}$), otherwise any traffic is allowed. Hence the probability P_{FP-N} is as follows:

$$\begin{aligned} P_{FP-N} = \; & P\{manual|non\_manual\} \\ & \cdot P\{non\_human|non\_human\} \\ = \; & (1 - R_{non\_manual}) \cdot R_{human} \end{aligned} \quad (3)$$

The second scenario (FP-M) happens when FIAT incorrectly blocks manual traffic (unpredictable manual events) – despite being correctly classified ($P\{manual|manual\}$) – because the human activity is incorrectly validated ($P\{non\_human|human\}$).

$$\begin{aligned} P_{FP-M} &= P\{manual|manual\} \cdot P\{non\_human|human\} \\ &= R_{manual} \cdot (1 - R_{human}) \end{aligned} \quad (4)$$

False negatives (FN) happens when the unpredictable manual event is incorrectly classified as control/automated ($P\{non\_manual|manual\}$), or it is correctly classified ($P\{manual|manual\}$) but the human behavior is incorrectly validated ($P\{human|non\_human\}$). Therefore, the probability of false negatives is

$$\begin{aligned} P_{FN} &= P\{non\_manual|manual\} \\ &\quad + P\{manual|manual\} \cdot P\{human|non\_human\} \\ &= 1 - R_{manual} + R_{manual} \cdot (1 - R_{non\_human}) \end{aligned} \quad (5)$$

Table 8: Testbed and experiments description. "*" refers that the devices are included in Mon(IoT)r dataset but no specific product version is noted, e.g., generation 3 or 4 for Echo Dot.

Device	Popularity Ranking		Presence in Public Dataset	
	Amazon	Best Buy	YourThings	Mon(IoT)r
Echo Dot 4	1	2	No	Yes*
Home Mini	-	7	Yes	Yes
WyzeCam	4	7	No	No
SP10	-	-	No	No
Home	-	-	Yes	Yes
Nest-E	3	3	Yes	Yes
Echo Dot 3	2	1	Yes	Yes*
E4 Mop Robot	14	-	No	No
Blink Camera	1	1	No	Yes
WP3	-	-	No	No

B MORE DETAILS ABOUT OUR TESTBED

Our testbed is composed of devices we had at our disposal. To demonstrate the representatives of the devices, Table 8 presents the popularity rankings (as in October 2022) of the devices in the corresponding category on both *amazon.com* and *bestbuy.com*. Overall, most of the devices are among the top-ranked devices and hence are representative to an extent, particularly the more complex and prevalent products such as smart speakers. It is noteworthy that the rankings might be biased because of the marketing decisions of the device vendors and the conflict of interests of the platforms, e.g., Google Home and Home Mini are not sold on amazon.com. It is also noteworthy that some devices may not be top-ranked because of being older. For instance, Google Home was first introduced back in 2016 while Echo Dot 3 and 4 were released 2 and 4 years after, respectively.

Table 8 further presents whether the devices in our testbed are also included in the public datasets of YourThings and Mon(IoT)r. We do not compare the predictabilities of the devices between the public datasets and our testbed because of the lack of accurate information about the frequency of the automated events and manual operations in the public datasets.