COTS Drone Detection using Video Streaming Characteristics

Anas Alsoliman
University of California, Irvine
United State
aalsolim@uci.edu

Giulio Rigoni
University of Florence
Italy
giulio.rigoni@unifi.it

Marco Levorato
University of California, Irvine
United State
levorato@uci.edu

Cristina M. Pinotti
University of Perugia
Italy
cristina.pinotti@unipg.it

Nils Ole Tippenhauer
CISPA Helmholtz Center for
Information Security
Germany
tippenhauer@cispa.saarland

Mauro Conti
University of Padua
Italy
conti@math.unipd.it

ABSTRACT

Cheap commercial off-the-shelf (COTS) drones have become widely available for consumers in recent years. Unfortunately, they also provide low-cost capabilities for attackers. Therefore, effective methods to detect the presence of non-cooperating rogue drones within a restricted area are highly required. Approaches based on detection of control traffic have been proposed but were not yet shown to work against other benign traffic, such as that generated by wireless security cameras. In this work, we propose a novel drone detection framework based on a Random Forest classification model. In essence, the framework leverages specific patterns in video traffic transmitted by drones. The patterns consist of repetitive synchronization packets (denoted as pivots) which we use as features in the proposed machine learning classifier. We show that our framework can achieve up to 99% detection accuracy over an encrypted WiFi channel using only 20 packets originated from the drone. Our system is able to identify drone transmissions even among very similar WiFi transmission (such as a security camera video stream) and in a noisy scenario with background traffic.

1 INTRODUCTION

The widespread availability of commercial drones and their use in a myriad of applications pose serious security and privacy concerns. For instance, unauthorized drones may be operated in restricted or crowded areas such as airports and stadiums, where they can collide with airplanes or land/crash on people. Moreover, most Commercial-off-the-Shelf (COTS) drones are equipped with cameras to enable First-Person View (FPV), which allows real-time video transmission from the drone's camera back to the controller (or any separate viewing device). Intuitively, this feature can easily lead to privacy violations. These issues motivated a surge of research efforts whose objective is the detection of unauthorized drones entering restricted airspace or private areas. Toward this goal, many recent contributions (*e.g.*, [6, 14]) detect drones by analyzing their radio signals.

The majority of COTS drones employ WiFi chips that allow the drone to act as WiFi Access Point (AP). The user can install an app in a smartphone/tablet to connect to the AP, and establish a bidirectional data stream to receive the FPV video stream and control the drone. Most prior work uses the FPV stream and analyzes traffic patterns from a statistical perspective. For instance, in [14] the authors presented a framework to detect drones' FPV streams based on channel use. That approach proved challenging, as video bit rates can widely vary in response to the specific captured area and drone motion. The authors also use the Received Signal Strength Indicator (RSSI) as a feature to discern moving vs. stationary devices. However, this latter approach was unable to differentiate drones from other moving radio sources. In [6], the authors presented a framework that used features extracted from WiFi frames exchanged between the drone and its controllers. Similar to the previously mentioned paper, the approach is not tested with changing bit rates emitted over the FPV channel and may be unable to differentiate drones' FPV streams from other WiFi video streams. Furthermore, the framework also required to have both the drone and its controller in the detection range. In [4], the authors trained a machine learning model to detect drones with high accuracy using at least 50 captured packets with a training set composed of drone and controller traffic. Despite using controller traffic in the dataset, the authors left the problems of "recognition of new UAV types" and "modified video patterns" as open problems.

In this work, we propose a novel drone detection framework that overcomes these shortcomings by leveraging specific packets we identified in the encrypted FPV stream sent by drones over the WiFi channel. We refer to these specific packets, which appear in video streams at periodic intervals for synchronization purposes, as "pivots". We use pivot patterns to build drone features and train a Random Forest machine learning model. The model is expected to produce high drone detection accuracy under challenging settings, including (i) the scene being captured by the drone's camera is highly dynamic which produces varying bit-rates, (ii) the drone, but not its controller, is within the packet capture range, (iii) the network traffic of the drone is completely encrypted, and (iv) there are other IoT devices in the environment actively emitting video streams that exhibit a FPV packets pattern similar to those generated by drones.

Under the aforementioned constraints, we can summarize our contributions as follows: (i) We introduce the concept of a *Pivot* as well as a *Fast Pivot Estimation* algorithm that quickly searches a stream of 20-packets to identify and extract pivots for each detected

This work was partially supported by Project *NALP-SAPR2: Navigazione Autonoma, Logistica, e agricoltura di Precisione per Sistemi Aeromobili a Pilotaggio Remoto e Robot*, granted by Fondo Ricerca di Base, 2019, University of Perugia.
This work was partially supported by *NSF under Grant IIS-1724331 and MLWiNS-2003237*.

device. (ii) We provide an implementation of the detection framework based on a Random Forest algorithm using features created from pivot packets with accuracy near 100%. (iii) We demonstrate that our solution detects drones the model has never trained on.

1.1 Related Works

Drones detection is a research topic which is capturing considerable attention from the research community. Mainly, drone detection techniques fall under four different categories: (i) Radar-Based Detection [19]: based on active radars that send electromagnetic pulses toward the area to be monitored to detect the electromagnetic energy reflected by any flying objects. (ii) Vision-Based Detection: based on computer vision devices (such as regular or Thermal cameras) to detect flying objects [11, 16]. (iii) Acoustic Detection: that detects the high-pitched sound frequency generated by the rotating propellers of the drone [7, 8]. And lastly (iv) RF/WiFi Detection techniques that monitor the wireless communication links between the drone and its controller (the Ground Control Station, or GCS) [15] [14]. Since our paper proposes a new WiFi detection technique, we focus on related works on RF/WiFi detection of drones. We also summarize some previous results on detection of WiFi cameras due to the similarities between the streams transmitted by the drones and those transmitted by fixed WiFi cameras.

Detection of Drone Traffic. The authors of [14] propose detecting drones by their FPV streams. The bit rate of a FPV stream emitted by a drone is compared with the bit rate of a well known and previously recorded FPV data streams. However, the more the scene changes, the higher the bit rate would be, and a drone recording a highly dynamic scene might not always match any known FPV bit rates. The authors also argue that since the drones move, the RSSI of their FPV channels would be different from those of other WiFi video streaming services. However, they did not take into account video streaming devices that might be moving such as VoIP applications on smartphones.

In [6], drones are detected by monitoring their FPV stream sent over WiFi and applying Machine Learning models. They extracted features from WiFi frames exchanged between the drone and its controllers. As in [14], the authors do not consider either the problem of detecting drones FPV streams with changing bit rates or that of differentiating between drones FPV streams and other WiFi video streams. The same authors, in [5], took the previous work a step further by improving the robustness of the algorithm. Precisely, they aim to detect a drone flying in stealth mode (*i.e.,* a drone that is not transmitting video).

In [12], a framework is proposed to fingerprint drone WiFi communications for the identification of specific drone models. Firstly, drones are discerned by other devices via their speed (exploiting the signal strength information used to calculate the acceleration). Once a WiFi session is associated with a drone, it is further classified using different features (*i.e.,* pattern of probe messages and information in a Frame Header). However, this detection is based on the assumption that other devices cannot move at the same speed as the drone. In some scenarios, a device could be inside a vehicle (*i.e.,* phone inside a car) that moves at comparable or higher speeds than of a drone.

In [17, 18], standard classification algorithms are used on eavesdropped traffic exchanged between a drone and its remote controller to analyze extracted features such as packets' inter-arrival times and sizes. The main aim is to detect a drone and its status *i.e.,* flying vs. resting.

In [4], Alipour et al. use classical features as those used in [17, 18] for differentiating FPV drones from other devices, and hence for detecting drones. The authors show that their framework detects with high accuracy drones using features extracted from packet samples of at least 50 packets exchanged between the drone and controller. The authors left the problems of "recognition of new UAV types" and "modified video patterns" as open problems. Our work aims to improve the results in [4] and to close the problem of recognizing new UAV types.

Detection of WiFi Cameras. In [9], the authors detect hidden wireless cameras using smartphones. Their system, DeWiCam, automatically analyzes wireless traffic for recognizing camera transmissions, specifically relying on physical and MAC layer features. Importantly, camera traffic streams have relatively stable volume and packet size pattern. Besides, wireless cameras can work continuously without interruptions in the stream.

In [13], the main objective is to identify hidden WiFi cameras by altering the ambient light captured by the cameras to induce variations in the camera's packet flow (caused by the video compression algorithm) which can be identified by statistical techniques. Both papers exploit the compression mechanism that cameras use for video transmissions, discussed below in Section 2. In our work, we are also interested in investigating characteristics of video transmissions, but for detecting drones under challenging scenarios.

1.2 Contribution

Our main contribution is the ability to detect drones for which the classifier has not been trained on while other video streaming devices exist in the environment. Our paper shows that it is possible to detect a new drone type by just extracting three specific features, defined by exploiting the characteristics of the drone traffic, from samples of 20-packets. Such specific features will be referred from now on as *unique-features*. The unique-features are based on properties of the size and inter-arrival time of the packets transmitted by the drone. Note that it is not possible to accurately detect a new drone type by just using standard statistical features similar to those adopted in [4]. Using the same specific three features, we can achieve overall detection with high accuracy (i.e., detection of classified drones) by just using 20 packets. Note that for the overall detection, the previous classifier proposed in [4] achieved a lower accuracy compared to the one we propose using a larger number of packets (i.e., 50). Additionally, our framework works even when

Table 1: Characteristics of the proposed framework.

	Alipour et al. [4]	Proposed Framework
Packet Sample Size	50	20
Captured Packet Flow	Drone + Controller	Drone
WiFi Capture Mode	Promiscuous	Monitor
Extracted Packet Features	Packet Size + Inter-Arrival	Packet Size + Inter-Arrival
ML Features Used	Statistical	Pivots
Detecting New Drones	No	Yes

the drone's controller is not in the monitored area. In summary, the method we propose is faster, more accurate than the previous classifier based on WiFi monitoring, and can detect new drone types on which the model has not been trained on, thus solving an open problem posed in [4]. Table 1 summarizes the differences between our framework and [4].

The paper is organized as follows. Section 2 provides an overview of the properties of the communication traffic involved in our scenario. In section 3 we give an introduction into our uniquely defined features, a novel algorithm for extracting these features, and a comparative analysis of the features behavior between FPV drones and other video streaming devices. Section 4 describes the data collection phase, our packet processing strategy, feature extraction, and the implementation of our Random Forest classifier. Field experiments and a thorough evaluation are presented in Section 5, while Section 6 concludes the paper.

2 BACKGROUND AND MOTIVATION

Our main objective is to detect drones entering a restricted area defined under the threat model described in Section 2.1. The detection process is accomplished by listening to the wireless channel. When a First-Person View (FPV) drone operates, a bi-directional connection is established between the drone and its controller. The downlink channel is a video stream sent by the drone to the controller, while the uplink channel carries control commands sent from the controller to the drone. Each channel has its own characteristics that will be described in Section 2.2 and 2.3 respectively.

2.1 Attacker and Defender Model

We consider a restricted area within which unauthorized drones need to be detected as soon as possible. We also assume that the drone controller might be outside of the monitored area (Figure 1). We do not use optical or audio measurements, but only radio spectrum monitoring using nodes deployed in the area. Different from prior work, we consider that other IoT devices such as cameras, that produce real-time video over WiFi might be present in the restricted area.

2.2 Video Encoding & FPV Channel:

The FPV channel transports a compressed video from the drone to the controller. A video is a stream or a sequence of still pictures. In order to decrease the portion of channel capacity used to transport the video over wireless channels, almost all video encoders include a compression algorithm.

Typically, compression algorithms exploit the – possibly high – correlation along the spatial and temporal dimensions within Group of Pictures (GoP). In brief, while JPEG compression is used in the spatial domain, the core idea to harness temporal correlation is to encode differences compared to reference frames within each GoP. We, then, have 3 frame types: Intra-Coded Frames (I-Frames), Predicted Frames (P-Frames), and Bi-directional Frames (B-Frames). I-Frames are Intra-coded frames that exploit spatial redundancy (correlation among the pixels in the frame) to achieve compression at individual frame-level. P-Frames and B-Frames are Inter-coded frames that exploit temporal redundancy prediction. P-Frames only encode pixels that are changed compared to the last reference frame, while B-Frame considers both previous and future reference frames. These frames are grouped into a single GoP, which starts and ends with an I-Frame.

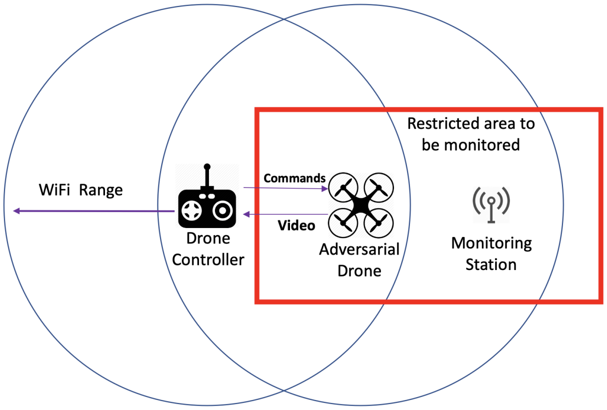


Figure 1: Attacker and Defender Model

Intuitively, scene characteristics and the motion of the drone heavily influence the compression rate achieved by the scheme described above, as well as the temporal structure of the data stream. If the captured scene is dynamic, then, the encoding scheme will generate I-Frames more frequently, which in turn results in shorter GoPs or larger P-/B-Frames, that is, a decreased compression rate.

Typically, video stream applications rely on well-known video codecs such as H.264 to compress raw images captured by some camera and turn them into compressed frames for faster transmission. In general, B-Frames are used on prerecorded videos and real-time streaming applications often do not use B-Frames as this would introduce a delay in the stream, and instead, only use P and I-Frames. The size of these frames is larger than a WiFi MTU (Maximum Transmission Unit) which is 2304 bytes. Network interfaces translate all packets sent and received from the OS into Ethernet, which has an MTU of 1500 bytes. Therefore, each individual video frame that is larger than the MTU is fragmented into a series of packets of size equal to the maximum MTU size, except for the last packet, which contains the residual of the frame.

FPV video streams also include other messages – such as synchronization information for maintaining the state of the channel. Typically, these messages are small and fit a single WiFi packet and are sent periodically at predicted intervals. Packets from the compressed video stream are emitted more frequently compared to sync packets, which then, have a higher probability to be buffered right after the last less-than-MTU sized frame packet.

The resulting pattern contains rather unique packet sequences. The fixed-size of the sync packets, make them suitable to act as **pivots** which are used to build FPV drone features for a detection model.

2.3 Control Channel:

The control channel carries flight commands from the controller to the drone. Overall, the control channel comprises of 2 different network flows, periodic control packets and "*heartbeats*" packets. In case of controllers that receive a video stream over TCP, a flow of acknowledgments (ACKs) will be sent out back to the drone as well. Note that even if ACKs are encrypted, they can be easily identified by the packet size (20 bytes TCP header without options + 20 bytes IP header without payload). Some control applications embed heartbeat messages within the stream of periodic control packets. Both streams (heartbeats and control packets) have rather unique sizes and inter-arrival times. Therefore, a simple detection system can easily differentiate between the controller's traffic and other background traffic. In our framework, we reflect a more realistic scenario where we assume that controllers are not always present within the range of the detection system and their traffic cannot be captured and classified (Figure 1).

Motivation: Intuitively, the packet pattern generated over the control channel is quite unique: short periodic packets. Although this approach can lead to robust detectors [5, 6, 10, 17], the main source of those packets is the controller, which might be out of range of the monitoring station/s (Figure 1). On the other hand, approaches focused on the FPV channel analyze statistical properties of this data stream. Specifically, they look at the timing and size of I- and P-Frames [9, 14]. However, drones generate data streams that are statistically similar to those of other IoT devices emitting video streams. Moreover, those characteristics widely vary as environmental and trajectory parameters change. We contend, thus, that many existing approaches would fail to provide robust detection in many relevant conditions. Therefore, the framework we propose is based on traffic emitted over the FPV channel to detect the drone even when the controller is out of the framework detection range and uses features robust to a wide spectrum of scenarios and parameters.

3 THE PIVOT APPROACH

One of the main contributions of the paper is introducing the *pivot* concept, which is a special packet found in the drone's FPV traffic which will be utilized as a drone feature. In this section, we first analyze the FPV stream emitted by the drone in order to define what exactly is the pivot and how it appears within the FPV stream. Note that, defining the pivot, we have access to the packet content. Secondly, once the pivot has been defined, we introduce the *Fast Pivot Estimation* algorithm that searches in linear time the stream of encrypted packets for potential pivots. Note that the fast pivoting algorithm works on encrypted streams and hence must discover the pivot without exploiting the packet content.

To define the pivot, we analyzed data collected from 3 FPV drones from 3 different brands: the EACHINE E58 WiFi FPV Quadcopter, Spacekey DC 014 FPV WiFi Drone, and Ryze Tello Quadcopter Drone. In the following, the 3 drones are referred to (according to their paint color) as Black (BLK), Red (RED), and White (WHT) drone, respectively.

Each drone has a First-Person-View (FPV) capability. Upon powering up, the drone creates an Access Point (AP) and the user connects his smartphone/tablet to that AP. Each drone has its own app that can be downloaded from Google Play and Apple's App Store. When the user connects to the drone's AP, a video stream can be received and viewed on the smartphone through the drone's designated app.

For every one of the 3 drones, we captured decrypted network traffic (for easier initial analysis) of the drone's FPV using an external WiFi adapter set into Monitor Mode from 5 meters away for 30 seconds. We consider 2 different video states: the first state corresponds to the drone's camera capturing a stable scene (STB), and the second state corresponds to a highly unstable (shaking) drone flight (SHK) where the camera rapidly points in different directions. The latter state induces fast changes in the captured video stream.

Pivot Definition via FPV Traffic Analysis: For the BLK drone, we observed that before the drone transmits a series of full-sized packets (which we assume is a video frame), the drone transmits a 46-bytes packet. This packet includes an ASCII-readable command called *lewei_cmd*, and it appears to be responsible for transferring media files from the drone to its associated app on the controlling smartphone [20]. The app associated with the BLK drone also sends some lewei_cmds to the drone, but only a few of them once every second. The RED drone, instead, sends 4-bytes packets (with identical payloads) to its RED-app right before sending a video frame. The RED-app sends the same 4-byte packet to the drone, but only 4 times in the entire 30 seconds network trace. Finally, the WHT drone sends an identical 35-bytes packet (except for the last 2 bytes in the payload) to its WHT-app between some video frames. These 35-bytes packets are sent at uniform intervals (every 0.1 of a second) which is independent of the varying FPV frames transmission times. This behavior was observed for both SHK and STB video states for all of the 3 drones.

We leverage these packets, which are repetitive and identical in length for each type of drone, and we call them the *pivot* packets that will be used to create our specific features.

Behavior of Pivot: We now study the appearance of the pivot and how it relates to the two different video states: SHK and STB. For each drone, for both the shaking state (SHK) and the stable state (STB), we computed the average bit-rate and the pivot appearance rate (pivot per second). The results are compiled in Table 2.

As one can see in Table 2, when the video state went from STB to SHK, the FPV bit-rate of the BLK drone on average has increased by 55 kbps (25%), while the number of pivots observed per second has increased by 2 (25%). For the RED drone, instead, the FPV bit-rate has increased by 4 kbps (2%), while the number of pivots observed per second has increased by 2 (13%). For the WHT drone, although the FPV bit-rate has increased by 42 kbps (13%), the number of observed pivots has remained almost constant.

Although the collected data from the BLK drone seems to support a dependence of the number of pivots on the FPV bit-rate (both increased by almost the same percentage), the behavior of the RED and WHT drones instead supports the hypothesis that the pivots are not necessarily tied to the bit-rate. Especially for the WHT drone since the inter-arrivals of the pivot packets are constant (10 seconds per packet), the number of pivot packets are almost the same for both the video states, independently of the bit-rate increase. The deviation in the number of pivots in the other two drones might be due to (i) the nonuniform inter-arrivals of pivots and (ii) the inherit packet loss in the wireless environment for packets captured via WiFi Monitor Mode. To this end, we can safely assume that the pivots are not necessarily tied to the bit-rate (*i.e.,* number of video

Table 2: Pivot frequency & bit-rate change

Drone	FPV Bit-Rate	Pivot/Second	Δ FPV Bit-Rate	Δ Pivot/Second
BLK-SHK BLK-STB	269.567 kbps 214.130 kbps	9.080 7.308	55.437 kbps	≈ 2
RED-SHK RED-STB	215.342 kbps 211.214 kbps	18.379 16.200	4.128 kbps	≈ 2
WHT-SHK WHT-STB	360.968 kbps 318.871 kbps	7.701 7.760	42.097 kbps	≈ 0

frames) of the FPV stream and, as such, the pivots can provide a new angle for feature extraction that will lead to drone detection.

3.1 Fast Pivot Estimation Algorithm

Although the pivot packets are repetitive packets that are sent within an FPV stream, and since the stream is assumed to be encrypted, pivots must be identified somehow. One possibility is to use their payload size. As different drones (even beyond the ones we considered) may have different pivot packets' structure, the challenge is how to discover the size of the pivot packets from the network trace itself. For this, we exploit the observation that pivot packets are sent more frequently compared to the other packet types. Based on this observation, we computed the occurrence frequency of packet sizes within drones' traces. We observed that the highest occurring packet size is the maximum packet size (MTU) for all of the drones we considered. Then, we observed the second-highest occurring packet size for all three drones (BLK, RED, and WHT) is the size of the pivot packet in all the cases.

So our initial approach to discover the size of the pivot in an encrypted stream has been to equate the size of the pivot with the size of the second-highest occurring packet. However, the size of the second-highest occurring packet is not stable when the the number of collected packets is too small. So, we studied the minimum number of packets required for the size of the second most frequent packet to become stable and equal to the size of the pivot that we know from our experiments.

In order to determine such a minimum number of packets, we computed the occurrence frequency of packet sizes across different ranges of packets (from 2 packets up to 1500 packets). We then computed the Cumulative Distribution Function (CDF) of all packet ranges. From the CDF plot (Figure 2), we found out that at least 170 packets are needed to be collected in order to correctly identify the pivot packet size for a given FPV trace with a probability of 0.95. Hence, using the occurrence frequency as a pivot size estimation approach would require a long observation period (sample of at least 170 packets), which clashes with the need to quickly detect drones as they enter the monitored area. Therefore, it is not possible to quickly determine the pivot size by discovering which is the second-highest occurring packet.

We then passed to a different approach which pin-points pivots within short samples of packets (no more than 20 packets). The new approach utilizes the video frames within the FPV stream as a guiding pattern. As discussed in Section 2.2, video frames are packetized by the Network Interface Card (NIC) such that each packet size does not surpass the Maximum Transmission Unit (MTU) of the transmitting NIC. Typically, the WiFi protocol has an MTU of 2304 bytes whereas Ethernet has an MTU of 1500 bytes. Since Operating Systems tend to translate all sent and received packets to and from the WiFi card into Ethernet, all packets have to adhere to the Ethernet's MTU which is 1500 bytes. Therefore, video frames that are larger than 1500 bytes are packetized into a series of 1500 bytes packets where the last packet size is typically less than 1500 bytes.

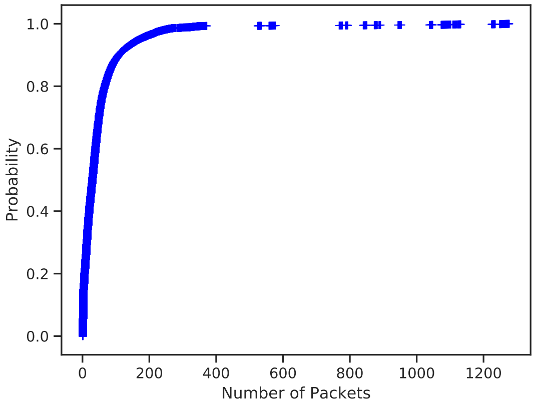


Figure 2: Cumulative Distribution Function of packets needed to acquire pivot size

Besides packetized video frames, in our scenario, the NIC also receives pivot packets from the drone application which, we recall, are synchronization packets sent periodically with large inter-arrivals between them (30 ms to 100 ms). On the other hand, each processed video frame is packetized and buffered at the NIC's transmission queue where each packet is sent out as soon as the wireless channel is cleared (inter-arrivals of packets belong to the same frame ≈ 0.01 ms). Therefore, there is a high probability that pivot packets are buffered between 2 packetized video frames. In other words, since each packetized video frame ends with a less-than-MTU packet and begins with an MTU packet, a pivot is most likely to appear after a less-than-MTU packet and before an MTU packet. From this observation, we established that a group of less-than-MTU packets between 2 MTU packets might include a pivot packet where the last packet is most likely the pivot. A pivot packet might also appear after 2 less-than-MTU packets or slip individually between 2 MTUs.

Based on these observations, we developed a *Fast Pivot Estimation Algorithm* that moves a sliding window across a stream of packets and records any continuous list of packets that have the following properties: (i) the list is less than 4 packets, (ii) the list is between 2 MTU packets, and (iii) each packet in the list has a size less-than-MTU.

Each recorded list of packets is denoted as a *pivot fingerprint* and might include a pivot packet (Figure 3). Therefore, there are 3 types of fingerprints: type 1 contains a single packet, type 2 contains 2 packets, and type 3 contains 3 packets. For each fingerprint, the last packet in the fingerprint is marked as a candidate pivot. From such candidate pivots, we can derive the pivot size.

It remains to explain how efficiently estimate the MTU packet size for a given sample of packets. The algorithm declares the size of the first packet in the sample as the MTU size and proceeds to find the next MTU packet. Whenever a packet size that is higher than MTU is detected, the Fast Pivot Estimation algorithm declares the new packet as the MTU and resumes the pivot estimation process.

Our Fast Pivot Estimation algorithm has correctly identified up to 95% of pivot packets within a packet trace in $O(n)$ time where n is the number of packets in the sample which in our framework would be 20 packets. Thus, the Fast Pivot Estimation can with high probability correctly identify the pivot size.

In the rest of this section, we analyze the video streams emitted by IoTs in order to compare them with the video streams emitted by FPV drones with respect to the pivots.

3.2 Pivots in Other Real-time Video Streaming

Real-time video streaming devices, such as IoT cameras, generate streams of packets that resemble a similarity to FPV drones' video streams. Namely, like FPV drones, video streams emitted by IoT cameras are unidirectional. That is, an IoT camera forwards a video stream to its designated app while the app maintains the video connectivity with the camera via heartbeats and keepalive messages. Note that this is unlike the video stream of VoIP applications (e.g. Skype), where the video streams are bidirectional.

To understand if our pivot approach could be able to distinguish IoT cameras from FPV drones, we collected video traces produced by 3 different IoT cameras: Wyze Cam, EZVIZ C1C, and Lefun MIPC. Just like the drones, we collected traces for each camera in a stable (STB) and dynamic motion (SHK) state, resulting in a total of 6 camera traces. We then calculated how many pivots were recorded per second and for each pivot type, in which type of fingerprint it was found.

In Figure 4 for each device (drone and camera), we computed the average number of pivots found per second for each fingerprint type. From the results, we can see that fingerprint type 3 has a very low potential for having a pivot in drones. On the other hand, each drone has at least 4 pivots per second that are located in fingerprint type 2. Both the Wyze Cam and Lefun MIPC cameras have some pivots in fingerprint types 2 and 3 ($\approx$ 2.5 pivots), but almost no pivots of fingerprint type 1 were detected in their packet traces. Instead, the EZVIZ camera in shaking state has on average about 20 pivots of fingerprints type 1, but almost no pivots of fingerprints type 2 and 3 in its packet trace.

From the results, we can conclude that other network devices that have network traffic patterns similar to drones might have different pivot patterns. Therefore, we are confident that the pivot approach will enable a robust drone detection.

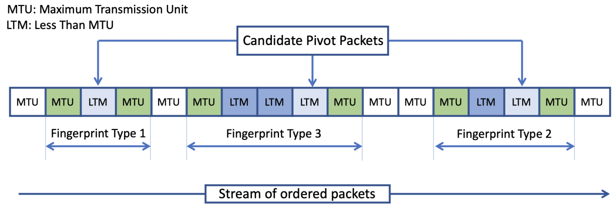


Figure 3: Candidate positions of pivot packets.

4 IMPLEMENTATION

In this section, we provide an overview of our setup, the devices that are used for generating network traffic, and the datasets created for our Machine Learning model. Then, we summarize the configuration parameters of the proposed Machine Learning model.

4.1 System Setup

To observe the traffic emitted by WiFi devices, we used an Acer Aspire F5-573G-759N laptop and Alfa AWUS036ACH WiFi adapter. We used Kali Linux 64-Bit version 2020.1 which was installed as a virtual machine using Oracle VirtualBox version 6.0.14. The WiFi adapter was set to Monitor Mode and tuned to the operating frequency of the AP being monitored. We used the Android app WiFi Analyzer [2] to find the correct operating frequency of the drone's AP. The network traces were collected using Wireshark [3] and parsed using Scapy [1].

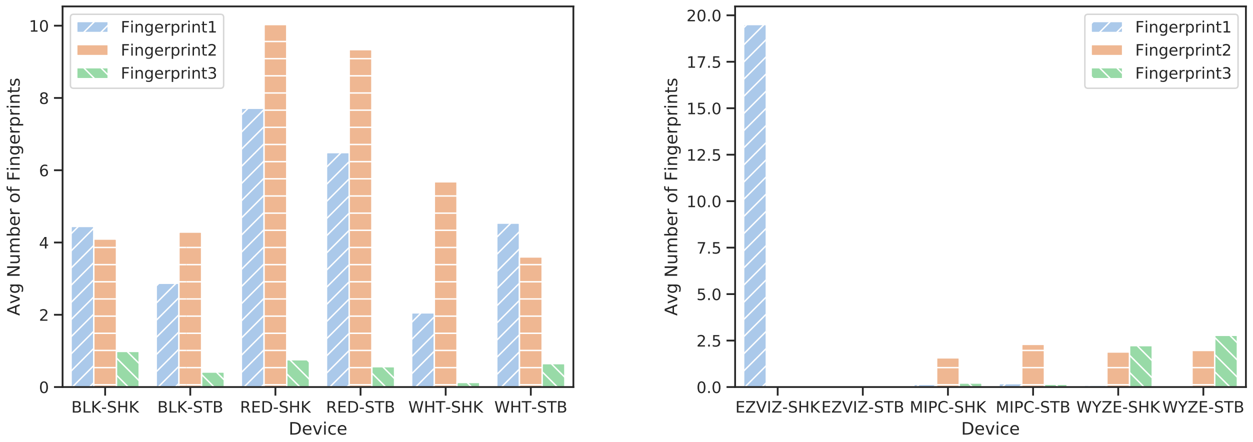
4.2 Data Collection

In the data collection campaign, we collected network traffic traces from different devices and applications under different conditions and scenarios. All data are collected over WiFi Monitor Mode (rather than on-device traffic capture) to include packet loss and other distortion effects created by real-world environments.

For each captured WiFi packet P_i transmitted by device D_j, we extract from the packet header the transmitter's MAC address, fragment number, sequence number, header flags, payload size $S_{i,j}$, and packet's arrival time $I_{i,j}$. We also extract the RSSI from the radiotap header that is added by the WiFi card upon receiving the packet. In case of fragmented packets, we used the sequence number, fragment number, and the "more fragments" flag in the header to put fragmented WiFi packets back into a complete packet. This was needed as WiFi adapters set into Monitor Mode do not reconstruct fragmented packets, and pass them individually to the operating system. The sequence number helped us identify retransmitted packets that were already captured. We keep track of the last 8 captured packets for each D_j and whenever a new packet is received, the new packet is checked whether it exists in the last 8 received packets. Note that in this way we partially solve the problem of the retransmitted packets. Namely, if a packet is retransmitted within 8 packets, it is discovered; otherwise, it is not. After capturing the WiFi packets, we reconstruct them if they are fragmented, then we group them as a list of packets based on the transmitter's MAC address such as $D_j = \{P_{1,j}, ..., P_{n,j}\}$. For each packet, we only save 1) the packet size and 2) its arrival time. Therefore, each device's packet list is transformed into $D_j = \{(S_{1,j}, I_{1,j}), ..., (S_{n,j}, I_{n,j})\}$ where $P_{i,j} = (S_{i,j}, I_{i,j})$. Then, we add each device list D_j to our Packet Dataset $DS_p = \{D_1, ..., D_m\}$.

The data collection strategy is then performed on the following devices and network services:

4.2.1 VoIP Apps. To test the performance of our framework, we ensured that our datasets include samples extracted from network devices that have traffic patterns similar to FPV drones'. In addition to the IoT camera traces collected in Section 3.2, we also collected traces of video traffic originated by Skype, Google Duo, and Discord VoIP apps while making video calls for 5 minutes. In Figure 5, we



(a) Drones Fingerprints.

(b) Cameras Fingerprints.

Figure 4: Average number of Fingerprint types per second for Drones (4a) and Cameras (4b).

can clearly see that bit-rates of drones, IoT cameras, and VoIP applications have similar traffic patterns.

4.2.2 *Non-Drone Background Devices.*

To ensure the completeness of our dataset, we collected 6 different traces of network traffic originated by network services for 3-minutes each. These traces are collected during file downloading, non real-time video streaming (YouTube), live video streaming (Twitch), Internet browsing, and social media browsing (Twitter and Facebook). The bit-rate of each application is plotted in Figure 5b.

4.2.3 *Drones & IoT cameras.*

The traces collected from drones and IoT cameras in section 3 are collected for initial analysis and are recorded under 2 different extreme cases, a completely stable video scene (STB) and a highly dynamic video scene (SHK). To include network traffic generated from a more realistic video pattern, we place all of the 3 drones and all of the 3 cameras in front of a screen playing a 5-minutes documentary video that contains slow, moderate, and fast-moving scenes. From each device, we captured the emitted video traffic which resulted in an additional 6 network traffic traces.

4.2.4 *Flying Drones.*

To further ensure that our dataset includes realistic samples, we recorded the FPV video of our drones while flying them. Before collecting the data, we performed an experiment to determine the maximum distance from the monitoring station at which a drone can be detected.

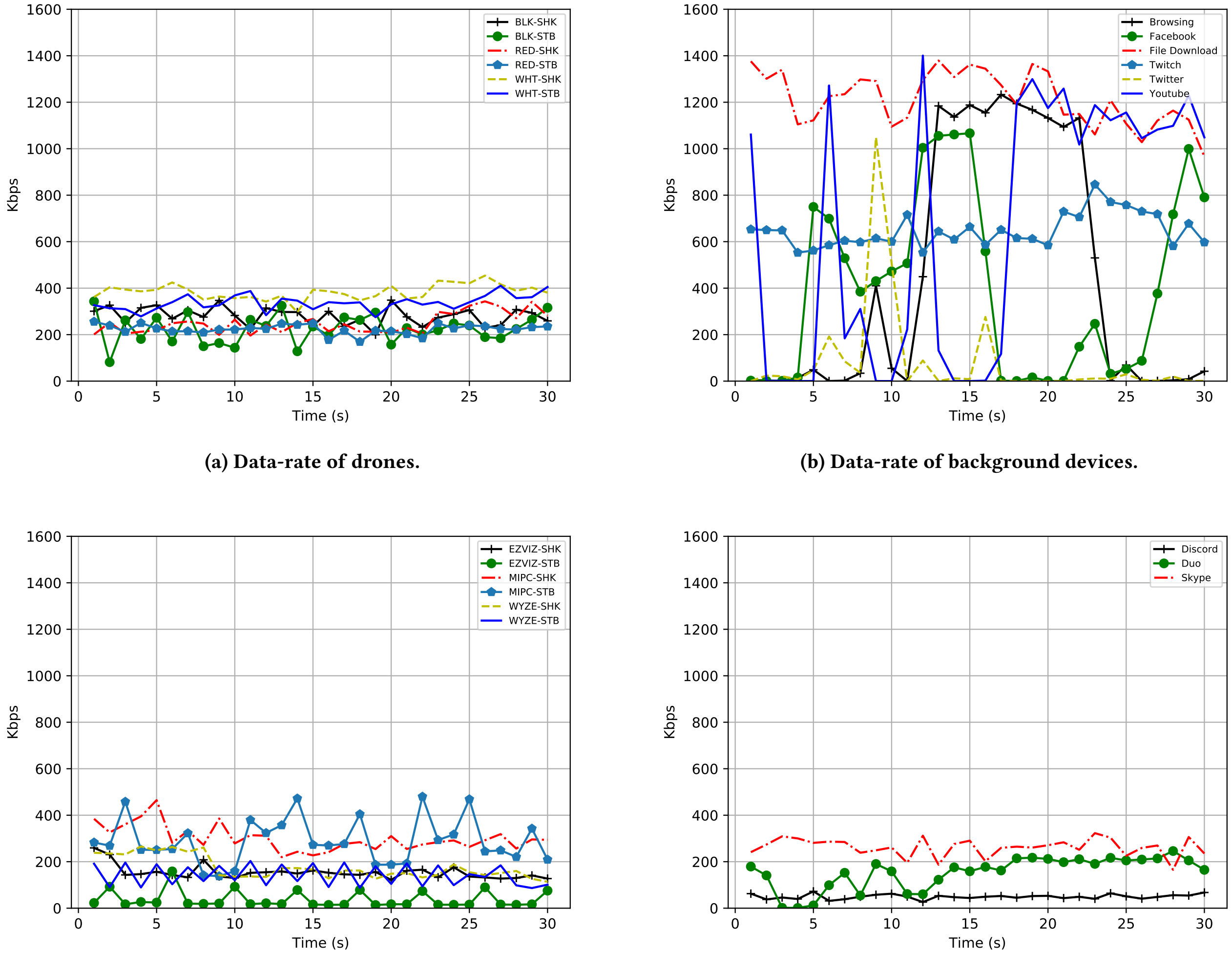
Detection Range Experiment: We set up a laptop as a monitoring station in the center of the university football field. Then we set up 7 waypoints in a straight line starting 10 meters from the monitoring station. The distance between each consecutive waypoint is 10 meters. We then flew each drone starting from the monitoring station in a straight line to the first waypoint while collecting the packets sent by the drone. Then, we flew the drone from the first waypoint to the second, up to the seventh while collecting the sent packets at each step. The drones we used are considered "micro-drones", and move at a relatively low speed with a rather small tilting of the drone body during motion.

Results: We noticed that the RSSI severely drops after 40 meters at waypoint 4 and the received power level at the monitoring station antenna is below −80 dBm. This is expected, as these drones are low-power devices with limited battery capacity. The packet loss between waypoint 3 and waypoint 4 was excessively high (around 70%). Therefore, we concluded that the effective detection distance for the micro-drones is 30 meters.

Data Collection from Flying Drones: We set up our monitoring station in the center of the football field and flown each drone for 3 minutes inside a detection area of radius equal to 30 meters. For each drone, we collected the emitted video traffic which resulted in 3 network traffic traces.

4.3 Feature Creation

From the collected packet traces DS_p from the previous section (Section 4.2), we created a series of 20-packets samples collected from each $D_j \in DS_p$. We used a sliding-window strategy with 8 packets offset where each sample interleaves with its adjacent sample by 8 packets such that the first sample from D_j is $SM_{1,j} = \{P_{1,j}, ..., P_{20,j}\}$ and $SM_{2,j} = \{P_{8,j}, ..., P_{28,j}\}$. Then for each sample, two sets of features are computed over the samples, statistical features (derived from [4], referred as *standard-features*) and features based on the pivot, referred as *unique-features*. Precisely, the unique-features are: (i) the computed pivot size, (ii) the ratio between the pivot size and the MTU size, and (iii) the ratio between the pivot size and the total sample size. Therefore, the samples of all devices are grouped together as a single list where each sample SM_i is transformed into a dataset record R_i to create the ML Dataset DS_{ML}. Each record $R_i = \{Ft_{pv_1}, ..., Ft_{pv_3}, \underbrace{Ft_{st_1}, ..., Ft_{st_{12}}}_{\text{packet size}}, \underbrace{Ft_{st_1}, ..., Ft_{st_{12}}}_{\text{inter-arrivals}}\}$ stores the three unique-features Ft_{pv_s}, with $1 \leq s \leq 3$, and the 12 standard-features Ft_{st_r}, with $1 \leq r \leq 12$ repeated twice. Namely,



(a) Data-rate of drones.

(b) Data-rate of background devices.

(c) Data-rate of cameras.

(d) Data-rate of VoIP.

Figure 5: Data-Rate for Drones, Background Devices, Cameras and VoIP applications over 30 seconds.

each standard feature is computed twice, once on the packet sizes and once on the inter-arrivals. The set of standard-features are described in Table 3 while the set of unique-features, based on the pivot, are described in Table 4.

4.4 Machine Learning Model

Our detection system is a ML-based system. The ML algorithm we adopt for the detection task is the Random Forest (RF). This is motivated by its low complexity and good performance in many settings. Moreover, RF is widely used in literature for similar problems.

Usually, ML algorithms require a training phase, a validation phase, and a final test phase. For each phase, a different chunk of the original dataset is used. Therefore, before we feed it to the ML algorithm, the dataset is divided into a training set for the training phase, and test set for the final test phase. We do not directly use a validation set because we employ a Cross Validation (CV) technique where we repeatedly divide the training set in a new training and validation set.

In the proposed detection system, we combine the RF algorithm with Grid Search; a technique where different hyperparameters are tested for a fine-tuning process. The hyperparameters that we used are: the maximum depth of the tree, the minimum number of samples required to split an internal node, the minimum number of samples required to be at a leaf node, and the number of trees in the forest. The basic idea is that the RF is repeated multiple times, with different hyperparameters, and the combinations of those hyperparameters that gives the best results in the validation set inside the Grid Search, are used for the final test using the test set.

Table 3: Standard-features calculated for both inter arrival time and size of packets.

Standard-features	Description
standard deviation	$\sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (x_i - mean(x))^2}$
variance	$\sigma = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - mean(x))^2$
root mean square	$\sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i)^2}$
m_square	$\frac{1}{N} \sum_{i=1}^{N} (x_i)^2$
p_skewness	$3(mean(x) - median(x))/\sigma$
kurtosys	$\frac{1}{N} \sum_{i=1}^{N} ((x_i - mean(x))/\sigma)^4$
skewness	$\frac{1}{N} \sum_{i=1}^{N} ((x_i - mean(x))/\sigma)^3$
min	$(Min(x_i)\vert_{i=1,\dots,N})$
max	$(Max(x_i)\vert_{i=1,\dots,N})$
mean	$\frac{1}{N} \sum_{i=1}^{N} (x_i)$
median	$\left\lceil \frac{N+1}{2} \right\rceil \vert (Sort_{i=1,\dots,N})$
medianAD	$median(\vert x_i - median(x)\vert)$

Table 4: Unique-features calculated from the pivot.

Unique-features	Description
pivot size	See Section 3.1
PM	pivot size/MTU size
PT	pivot size/total sample size

5 EVALUATION

In this Section, we assess and discuss the performance of our system, tested in different scenarios. In Section 5.1 we test the capability of our system to detect drones that have not been observed before, while in Section 5.2, we report the results in the overall detection, that is, the detection in a standard scenario with known drones.

5.1 Unknown Drones

As claimed, one of our goals is to test the performance of our system on detecting unknown drones. For this purpose, we tested our detection model under a scenario in which an unknown drone approaches our detection range. To prove the effectiveness of our system in this case, we trained our system as the following: we split the dataset DS_{ML} into a train and test set (as described in Section 4.4), but in this specific case the train set is composed only by the data of two drones and background traffic, while the test set is composed with the data from the remaining third drone mixed with background traffic. For example, to detect the WHT drone, we train our framework on the traffic of the RED and the BLK drones, completed with background traffic. We repeated this experiment for all the drones we have. The detection accuracy of each different drone is reported in Figure 6. For each unknown drone, we repeat the detection experiment using either the standard-features or our unique-features. The standard-features have good results for the RED and BLK drones, but poor accuracy for the WHT drone. Instead, the unique-features perform well for all the drones, and in particular for the WHT one. This shows that the unique-features are able to capture some specific properties of drones that, instead, the standard-features miss. The accuracy of the unique-features is always above 80%. Since the unique-features perform better than the standard-features in detecting a new drone for which the classifier was not trained, we can conclude that our unique-features provide a different angle at measuring drones' FPV traffic characteristics and contribute in advancing the open problem set in [4] which is detecting unknown drones.

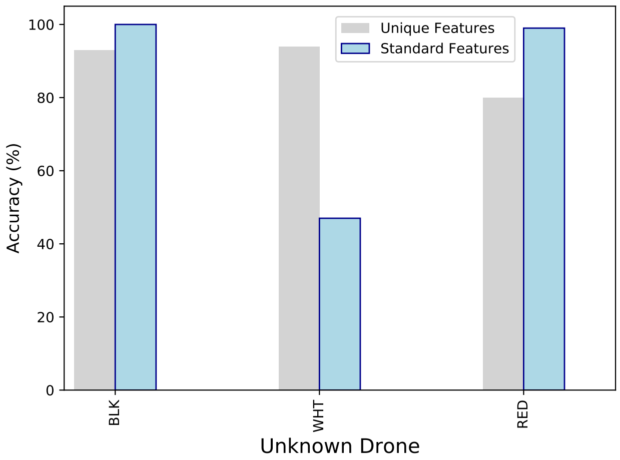


Figure 6: Accuracy in detecting an unknown drone with standard and unique-features.

5.2 Overall Detection

We also test the capability of our framework to detect drones that are known beforehand. For this experiment, we use a dataset composed of all the data we collected from drones and background traffic (*i.e.,* all the devices and applications presented in Section 4.2) that are randomly mixed together. As described in Section 4.4, we divided the dataset in training and test set, where the training set is roughly 80% of the whole dataset.

Using our unique-features based on pivots, the system achieves an accuracy of 98%, which is slightly less than the accuracy of 99% achieved using the standard-features. Nonetheless, our unique-features achieve overall better performance in detecting new drones, as shown in the previous Section 5.1, and so we believe that the loss of 1% in the accuracy can be overlooked.

It is worthy to note that these results show that the system is able to correctly discern almost all the drones' traffic from the background traffic in the test set. Moreover, being the background traffic made from various types of devices and applications that resemble similar traffic patterns to drones' (as shown in Figure 5 and explained in Section 4.2), our system is able to discern the drone traffic among traffic produced by VoIP applications and cameras. To better support this statement, we report the result of an ulterior test (Figure 7) in which we detect drones traffic between only camera traffic or VoIP traffic. In the VoIP traffic case, the dataset was heavily unbalanced due to the number of drone samples far greater compared to the number of VoIP samples. For this reason, only for this test, the drone samples are reduced in number (equally between

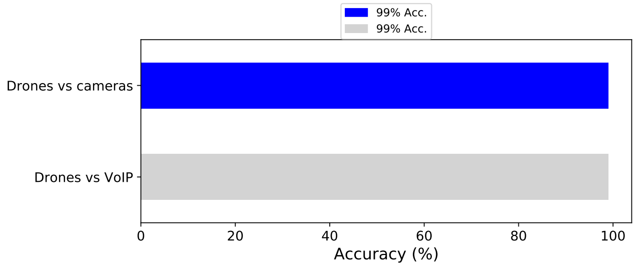


Figure 7: Accuracy in discerning drones traffic from camera traffic and VoIP traffic.

Table 5: Comparison of achieved accuracy.

	# Packets	Features	Accuracy
Proposed Framework	20	Standard- Unique-	99% 98%
Alipour et al. [4]	50	Standard	≈ 88%

the three drones) to match the VoIP samples, thus the dataset can be considered balanced.

Finally, we compared our results on overall detection with those, found in the literature, obtained by applying the state-of-the-art algorithm in [4]. Our framework using samples of 20-packets is able to overall discern the drones from similar applications with accuracy higher than 98% independently of the kind of features used. Instead, the state-of-the-art [4] using samples of 50 packets has an accuracy of 88%, which is lower than that achieved by our framework just using 20 packets (see Table 5). In conclusion, our framework works better and requires fewer packets than [4] when it is used to discover the presence of known drones.

6 CONCLUSIONS

In this paper, we proposed a COTS Drone detection framework solely based on FPV packets. In particular, we introduced the notion of “pivot”, that is, drone-specific synchronization packets that are interleaved with the video stream. The pivots are used as a guideline to identify unique-features that are then used to train a ML-based system built from a Random Forest classifier. We also proposed a *Fast Pivot Estimation* algorithm that quickly searches a sample of packets for pivots in linear time. Our experiments demonstrated that our unique-features are able to detect new drones. That is, the model is able to correctly classify drones whose features are not included in the model’s training set. Furthermore, our framework is able to detect new drones among other devices that exhibit similar behavior to drones’ FPV such as IoT cameras and VoIP video calls. Our implementation and evaluation of the framework demonstrated an overall detection accuracy up to 98% with our unique-features, for a maximal distance of 30m between the detector and the drone (without optimized antennas). We conclude that it is possible to detect non-cooperating third-party COTS drones with only 20 packets based on their FPV traffic with third-party controllers enabling operators of critical infrastructure (such as airports) to set up a perimeter for early detection. We address, then, a critical weakness of prior solutions, which require that, for every new drone released on the market, a new data collection and training phase is performed to include that drone in the dataset.

REFERENCES

[1] 2020-03-13. Scapy. https://scapy.net/

[2] 2020-03-13. Wifi Analyzer. https://play.google.com/store/apps/details?id=com.farproc.wifi.analyzer

[3] 2020-03-13. Wireshark. https://www.wireshark.org/

[4] Alipour-Fanid et al. 2019. Machine Learning-Based Delay-Aware UAV Detection and Operation Mode Identification over Encrypted Wi-Fi Traffic. *IEEE Transactions on Information Forensics and Security* 15 (2019), 2346–2360.

[5] Bisio et al. 2018. Improving WiFi Statistical Fingerprint-Based Detection Techniques Against UAV Stealth Attacks. In *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 1–6.

[6] Bisio et al. 2018. Unauthorized Amateur UAV Detection Based on WiFi Statistical Fingerprint Analysis. *IEEE Communications Magazine* 56, 4 (2018), 106–111.

[7] Busset et al. 2015. Detection and Tracking of Drones using Advanced Acoustic Cameras. In *Unmanned/Unattended Sensors and Sensor Networks XI; and Advanced Free-Space Optical Communication Techniques and Applications*, Vol. 9647. International Society for Optics and Photonics, 96470F.

[8] Ellen E Case, Anne M Zelnio, and Brian D Rigling. 2008. Low-cost Acoustic Array for Small UAV Detection and Tracking. In *2008 IEEE National Aerospace and Electronics Conference*. IEEE, 110–113.

[9] Yushi Cheng, Xiaoyu Ji, Tianyang Lu, and Wenyuan Xu. 2018. Dewicam: Detecting Hidden Wireless Cameras via Smartphones. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*. 1–13.

[10] Martins Ezuma, Fatih Erden, Chethan Kumar Anjinappa, Ozgur Ozdemir, and Ismail Guvenc. 2019. Micro-UAV Detection and Classification from RF Fingerprints Using Machine Learning Techniques. In *2019 IEEE Aerospace Conference*. IEEE, 1–13.

[11] Sai Ram Ganti and Yoohwan Kim. 2016. Implementation of Detection and Tracking Mechanism for Small UAS. In *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 1254–1260.

[12] Li et al. 2017. Drone Profiling through Wireless Fingerprinting. In *2017 IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*. IEEE, 858–863.

[13] Tian Liu, Ziyu Liu, Jun Huang, Rui Tan, and Zhen Tan. 2018. Detecting Wireless Spy Cameras via Stimulating and Probing. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. 243–255.

[14] Nassi et al. 2019. Drones’ Cryptanalysis-Smashing Cryptography with a Flicker. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1397–1414.

[15] Phuc Nguyen et al. 2017. Matthan: Drone Presence Detection by Identifying Physical Signatures in the Drone’s RF Communication. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. 211–224.

[16] Artem Rozantsev, Vincent Lepetit, and Pascal Fua. 2015. Flying Objects Detection From a Single Moving Camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 4128–4136.

[17] Sciancalepore et al. 2019. Detecting Drones Status via Encrypted Traffic Analysis. In *Proceedings of the ACM Workshop on Wireless Security and Machine Learning*. 67–72.

[18] Sciancalepore et al. 2019. Picking a Needle in a Haystack: Detecting Drones via Network Traffic Analysis. *arXiv preprint arXiv:1901.03535* (2019).

[19] SpotterRF. 2020-03-13. Drone Detection System. https://spotterrf.com/applications/drone-detection/

[20] Junia Valente and Alvaro A Cardenas. 2017. Understanding Security Threats in Consumer Drones through the Lens of the Discovery Quadcopter Family. In *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy*. 31–36.