Cybersecurity Analysis in the UAV Domain: the Practical Approach of the Labyrinth Project

Oliver Jung
Oliver.Jung@ait.ac.at
Austrian Institute of Technology (AIT)
Vienna, Austria

Abdelkader Magdy Shaaban
Abdelkader.Shaaban@ait.ac.at
Austrian Institute of Technology (AIT)
Vienna, Austria

Francesca Soro
Francesca.Soro@ait.ac.at
Austrian Institute of Technology (AIT)
Vienna, Austria

Miguel-Ángel Fas-Millán
MiguelAngel.FasMillan@dlr.de
Deutsche Zentrum für Luft- und Raumfahrt (DLR)
Weßling, Germany

ABSTRACT

In the last decades, Unmanned Aerial Vehicles (UAVs) are finding more and more fields of application. Their flexibility and cost-efficiency make them useful to support complex operations in agriculture, remote sensing or construction, just to name a few. In the Labyrinth project we aim at investigating the applicability of UAV usage to critical scenarios like air, water and road traffic control or emergency, with a strict focus on safety, security and efficiency. This involves also the cybersecurity aspect, which is the main focus of this work. UAVs used in critical applications are in fact potentially exposed to a wide set of cyber threats. The NIST cybersecurity framework [17] defines five different security functions which are: identify, protect, detect, respond and recover. In this paper we address the identify and detect functions with an approach involving threat analysis and anomaly detection. Firstly, we identify which threats pose a significant risk to the Labyrinth use case, for instance leading to the collision of UAVs in case an attacker is successful. Secondly, we present a machine learning-based pipeline aimed at detecting deviations in the position reportings of the drone, to support the detect function during flight operations. The pipeline is tailored to the Labyrinth system reporting needs and is based on unsupervised machine learning to overcome the lack of labeled data. Anomalous points, i.e., points deviating from a coherent path, potentially because of a cyber-attack or a failure, are visually separated from the coherent ones and marked as noise. To prove its robustness, we test the pipeline introducing artificial perturbations in the data.

CCS CONCEPTS

• **Security and privacy → Intrusion detection systems**; **Denial-of-service attacks**; • **Hardware → Emerging tools and methodologies**.

GoodIT '23, September 06–08, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0116-0/23/09.
https://doi.org/10.1145/3582515.3609566

KEYWORDS

Unmanned Aerial Vehicles, Anomaly Detection, Security Analysis, U-space, Unmanned Traffic Management

ACM Reference Format:
Oliver Jung, Abdelkader Magdy Shaaban, Francesca Soro, and Miguel-Ángel Fas-Millán. 2023. Cybersecurity Analysis in the UAV Domain: the Practical Approach of the Labyrinth Project. In *ACM International Conference on Information Technology for Social Good (GoodIT '23), September 06–08, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3582515.3609566

1 INTRODUCTION

The trend in the usage of Unmanned Aerial Vehicles (UAVs) has surged over the last decade and is constantly rising. Their flexibility and cost-effectiveness allow to find applications in multiple fields, such as disaster research and management [1], precision agriculture [22], remote sensing [14], building inspection [13] and even delivery [15], just to name some of the most used nowadays. In this context, the EU H2020 project Labyrinth[1] aimed at investigating UAV applications to enhance safety, security, and efficiency in civil transport and services; but always under the tutelage of an Unmanned Traffic Management system (UTM), which can be seen as an automation of the service provided by air traffic control in general aviation. This UTM system was designed following the guidelines of the U-space concept of operations (ConOps), the European harmonized approach to manage air traffic below 120 m altitude, where the small drones will operate. In the Labyrinth project, the U-space environment implemented was used to support four use cases representing different final users willing to test the integration of drones in their daily activities: i) road traffic control, including speed estimation, plate identification, or watching for violations such as driving while using the smartphone or not wearing a seatbelt; ii) waterborne transport, requesting drones for security surveillance of the port, monitoring and documentation of container loading operations, or to check dredge waste in the waters; iii) air transport, to be used in bird shepherding tasks; and iv) emergencies, to assist medical first responders in activities like inspection of the area of work to determine actions required and ways of access, deliver medical supplies or, with drones equipped with lights and a megaphone, guide people to follow a escape route.

[1] https://labyrinth2020.eu/the-project/

Use cases like the previous ones are exposed to cyberthreats which could lead to severe consequences for humans and the environment. In this paper, we present the experiences and lessons learned during the Labyrinth project, focusing on the practical aspects required to identify potential threats, and the actions to support their detection. The menaces identified target mainly the position information constantly reported by the drones to the UTM system during the flight; these attacks need to be detected in a timely manner so that appropriate measures can be taken by the UTM system (warning broadcast, delivering of tactical separation instructions) and the drone operator. As a first step to support the identification phase, we present a threat analysis executed using the ThreatGet tool [16] and refined according to the STRIDE methodology [8]. Secondly, to support the detect function during flight operations, we present a lightweight machine learning-based pipeline aimed at detecting deviations in the transmitted location data. The pipeline is tailored to both real-world and simulated data collected by Labyrinth's flight operator partners and relies on unsupervised machine learning to overcome the lack of a labeled dataset. During the flight, the UAVs reported their position once per second. Anomalous points, i.e., locations deviating from the expected trajectory, are visually separated from the coherent ones and marked as noise by the algorithm. To prove the robustness of our pipeline, we test the detection capabilities introducing artificial perturbations.

The remainder of this paper is organized as follows: Section 2 provides an overview of the related work; Section 3 introduces the architecture of the UTM environment developed in Labyrinth and the format of the collected data; Section 4 discusses the threat analysis methodology and results; Section 5 presents the anomaly detection pipeline and results; and Section 6 concludes the paper.

2 RELATED WORK

Many works targeting various types of UAV anomaly detection can already be found in the literature. UAVs are equipped with different kinds of sensors required for flight control - like GPS chips, as well as the so-called Internal Measurement Unit (IMU), consisting of accelerometers and gyroscopes to measure acceleration and rotation - or sensors needed for fulfilling its final task, like cameras or laser scanners. Moreover, the UAV communicates remotely with the Ground Control Station (GCS), which sends control commands to the UAV and receives location reports over communication links. Thus, for detecting attacks different types of information can be used.

A very relevant type of attack in the UAV domain is GPS spoofing [20]. GPS spoofing refers to different approaches to generating a fraudulent GPS signal so that the GPS receiver provides an incorrect position. GPS spoofing can even be used to hijack UAVs [18]. Panice et. al. propose in [12] an approach for detecting GPS spoofing attacks on UAVs based on the analysis of state estimation using a Support Vector Machine (SVM).

Other works focus on detecting anomalies in the records of various UAV sensors. Sun et al. [19] propose an Adaptive Neuron Fuzzy Inference System (ANFIS)-based approach for the detection of on-board navigation sensor faults in UAVs. They, however, require an offline database of labeled data to train the model. Authors of [5] use Structured Sparse Subspace Learning to detect anomalies in the navigation altitude in simulated data. In [21], Wang et al. apply Long Short Term Memory (LSTM) Recurrent Neural Network to real UAV sensor data points. This technique requires a training dataset of normal behaviors to be effective. Finally, Bu et al. [3] combine GNSS sensors and IMU sensors data to develop an integrated algorithm for detecting UAV on-board navigation sensor anomaly, by combining particle filter (PF) estimated state residuals with fuzzy inference system (FIS) decision system.

Attacks targeting the communication links between the UAV and the GCS, are often successful due to the lack of complex cryptographic measures, required for a high level of security. The reason for such lack is sometimes the limited processing power of the onboard computers and the battery economy. For this reason, some works propose the usage of lightweight intrusion detection and prevention system (IDPS) modules for UAVs. The one presented in [2] is trained using Deep Reinforcement Learning (DRL), specifically Deep Q-learning (DQN), to enable UAVs to autonomously detect suspicious activities and to take necessary action to ensure the security of the UAV communication networks and UAV sensor information. An intrusion detection system (IDS) based on a recurrent neural network using LSTM cells is used in [7] to analyze and detect suspicious sequences of Micro-Air-Vehicle communication (MAVlink) [9] message identifiers. The authors present moreover the implementation of application layer Denial of Service (DoS) attacks using MAVlink heartbeat flooding, ping flooding and MAVlink request flooding.

In our paper, the anomaly detection function is strictly related to the threat identification phase. The anomaly detection pipeline is tailored to the output of the threat analysis and to each use case's specificities. In the case of the data analyzed in this project, only timestamped, unlabelled location data are available, which makes some of the previously analyzed methodologies ineffective or non-applicable. The specific design of the use cases calls therefore for unsupervised, lightweight algorithms.

3 ARCHITECTURE AND SYSTEM MONITORING

In this Section we introduce the high-level system architecture of the Labyrinth environment. Figure 1 shows it as modeled by ThreatGet [16]. Its four main components are: the UAV, the GCS, the UTM cloud system, and the Bridge server. Table 1 describes the functionality of the different components. The diagram depicts them as nodes connected through different link communication options to perform their information exchanges. A mobile GCS would communicate with the UAV and the Bridge Server through wireless communication channels, while not necessarily of the same type. We could also have a stationary GCS connecting the Bridge via ADSL. Both Bridge and UTM are hosted in the same cloud infrastructure; this makes them independent of the connectivity option chosen by the operators: 4G, 5G, SATCOM, or ADSL.

It should be noted that all the components consist of different sub-components. In fact, following the U-space philosophy, some UTM services could be outsourced, provided by different, specialized companies. This was the case in Labyrinth of the path planner, which was provided by the partner Universidad Carlos III de Madrid (UC3M), while the rest of UTM services where hosted in DLR's

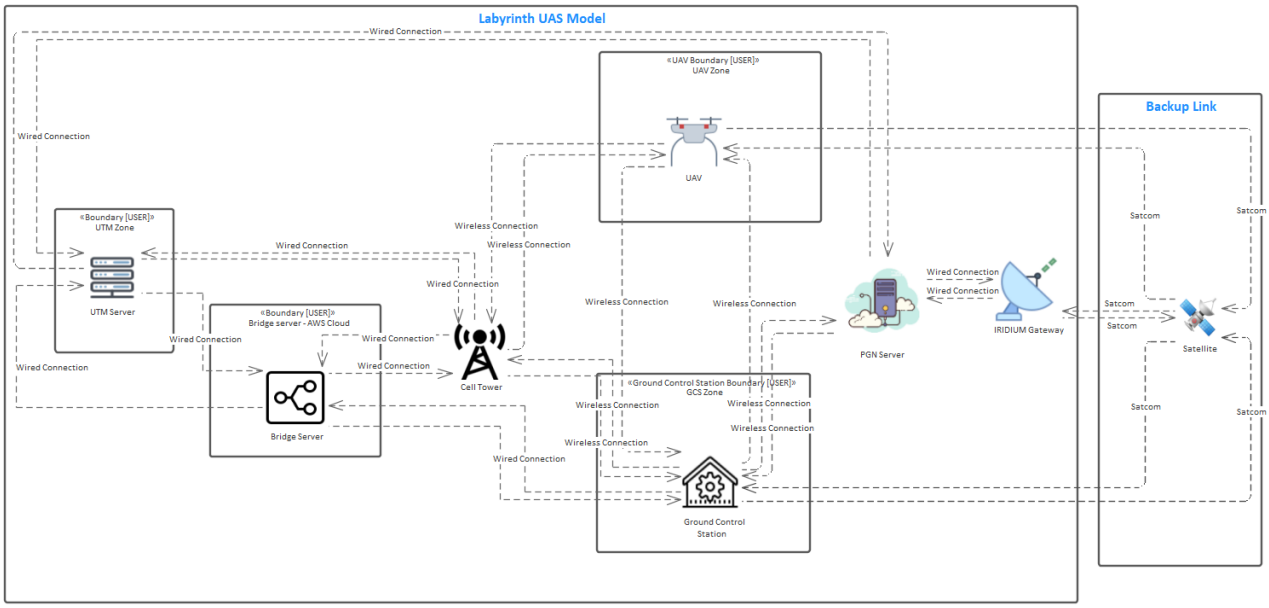


Figure 1: Labyrinth system high-level model.

cloud, from which the UC3M service was called when necessary. Anyway, regarding the application of the anomaly detection in the Labyrinth system, the sub-component architecture can be obviated, since the depicted high-level model is enough to grab the concept. However, according to IEC 62443 [6], which has been applied for defining security requirements, we use a slightly different terminology, since IEC 62433 uses the term *zone* for what has been called *component* in the following.

As can be seen in Figure 1, there are different points in the environment where data exchange can be monitored. Location and timestamp are packed together with other telemetry values like speed, altitude, accuracy or the drone and flight unique identifiers in the same report message type. Given the relevance of this information — especially for the UTM, since it takes decisions and broadcasts traffic information updates based on the UAV reports — it is important both to protect the integrity of this data and to detect possible deviations. The reasons for a deviation might be found in data tampering, spoofing, or technical failures of the sensors. The possibility of each event and its impact on the system needs to be identified during the threat analysis phase. Subsequently, our anomaly detection algorithm needs to detect said deviations. For the present work, the messages scanned in the anomaly detection phase are those received by the UTM from the GCS, but our algorithm could be also applied to the messages received by the UTM directly from the UAV, since their content should be exactly the same; operators decide if they prefer to report from the UAV, fromo the GCS of from both redundantly.

4 THREAT ANALYSIS

In the following, we performed a threat analysis for the information flows in between the different components with the aim to identify the most critical flows and components.

4.1 Methodology

The threat analysis is based on the usage of the ThreatGet tool, developed by AIT, together with the STRIDE model developed by Microsoft. ThreatGet employs a database of common cybersecurity threats in the UAV domain to understand how vulnerabilities and attacks could propagate across the system. We initially test the security of Labyrinth's high-level structure (see Figure 1) without applying any security mechanisms because we want to see potential security vulnerabilities in the system model, including components, assets, and communication channels. The outcome provides a better understanding of the system's cyber risks and determines the most suitable security measures to implement. We subsequently focus on more specific functionalities of the system, classified by means of the STRIDE model. This model is necessary to analyze how the individual threats affect the single components or assets and their connection to other components. A distinction is made between 6 categories of threats, explained in Table 3.

4.2 Results

4.2.1 ThreatGet High Level Findings. ThreatGet identifies 230 threats in the high-level Labyrinth system model. The output of the analysis is the so-called *Threat List*. A snippet of the list is reported in Figure 2. Some of the identified threats could impact multiple components and the applied security properties for each component in the system design; they could be propagated through the communication channels between interconnected components. For this reason, some threats could appear more than once in the Threat List. An example is threat - T66:73, "Protocol flooding for node bombarding", which impacts the GCS as single Target, but multiple Sources (UTM, Bridge, UAV, etc.).

Table 1: Labyrinth components and their functionality.

Component	Description
UAV	The UAV is able to support different kinds of applications. It is controlled via the GCS. Trajectories are uploaded to the UAV by the GCS. It sends location information directly to the UTM and to the GCS.
GCS	The UAV is controlled by the GCS. Trajectories are received from the UTM system and uploaded to the UAV for execution.
WebApp	Optional web interface of the UTM services. Provided to avoid operators modifying the GCS graphic interface to embed the UTM dialogues and information.
UTM	Conglomerate of coordinated services like strategic and tactical deconfliction, tracking, monitoring, flight plan processing or traffic information. Keeps the UAVs separated by providing 4D trajectories and geofence provision.
Bridge server	Required to coordinate the actions in the web with the associated message delivery from the UTM to the GCS. Also keeps updated each user view of a same operator. Hosts a drone simulator.

Table 2: STRIDE Categories.

Threat	Affected property	Definition
Spoofing	Authentication	Pretending to be something else or another person
Tampering	Integrity	Changing data or code
Repudiation	Non-Repudiation	Pretending not to have performed an action
Information Disclosure	Confidentiality	Exposing information to someone not authorized to see it
Denial of Service	Availability	Denial or degradation of quality of service to users
Elevation of Privileges	Authorization	Acquisition of rights without proper authority

4.2.2 Location Report STRIDE Analysis. We report here an example of STRIDE analysis focused on threats to location reporting. The analysis takes into account the reporting from the UAV to the GCS, from the UAV to the UTM, and from the GCS to the UTM. The results are reported in Table 3. From the Table, we recognize that the three types of location reporting practically share the same threats: the attacker is likely to Spoof or Tamper information and/or perform Information disclosure or Denial of service attacks. The lowest score is assigned to the Information disclosure threat, as the location of the UAV is generally a public information (in fact, the U-space ConOps contemplates the need of the drone *electronic conspicuity*, the need to be visible and identifiable by citizens, operators or other drones). The remaining threats are assigned an impact from 2 to 4 (2 to 3 in the case of Spoofing from UAV to UTM). The actual rating strongly depends on the situation in which the attack takes place, e.g., if the UAV exposed to a DoS attack is in an isolated area or not, or if the drone can receive tampered or spoofed commands from the attacker. In this latter case, the risk will be maximum.

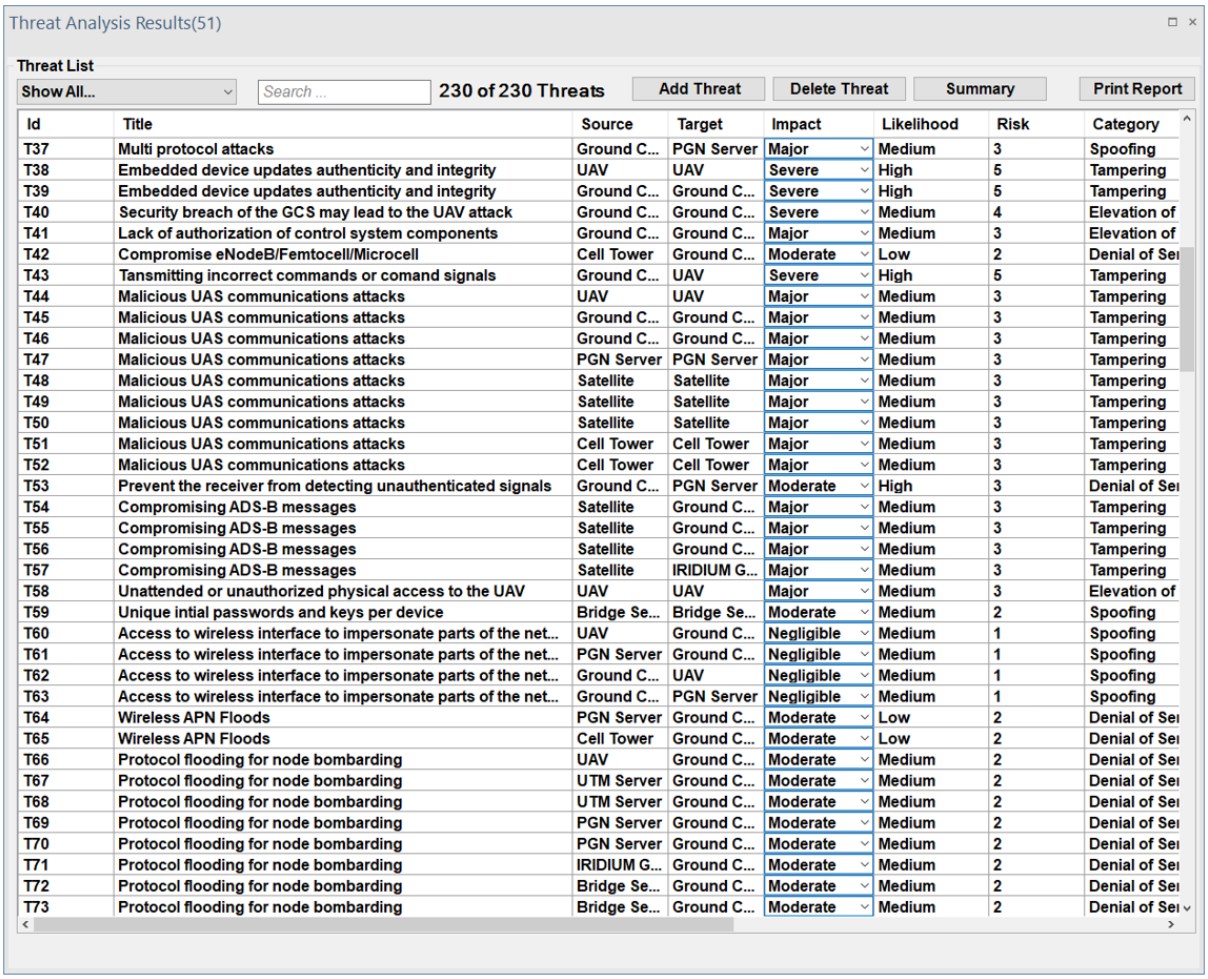
5 ANOMALY DETECTION

In this Section, we present the framework for anomaly detection that has been tested in the Labyrinth project. We develop a lightweight unsupervised Machine Learning framework to detect tampered or unavailable location measurements —- sent and collected during flight operations — within a time frame that can be tuned to guarantee the implementation of adequate security measures. We choose an unsupervised clustering algorithm, DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [4], to avoid the need for labeled and large datasets, as well as to avoid potentially long training phases.

5.1 Machine Learning Pipeline

The implemented pipeline follows the standard steps. We first acquire the spatial coordinates together with their timestamp; we then tune the algorithm parameters, proceed with the iterative clustering step and eventually verify the model robustness artificially introducing new anomalous points.

As previously stated, the final aim of this task is to provide an automatic framework to detect errors in the coordinate's transmissions, possibly coming from cyberattacks or defects in the communication channel. Given the unavailability of labelled data, we choose to apply DBSCAN, a density-based clustering algorithm. DBSCAN is designed to identify clusters of data points in a dataset based on their density. Unlike traditional clustering algorithms like K-means [10], DBSCAN does not require a pre-defined number of clusters. Instead, it groups together data points that are close to each other and have sufficient density while considering other data points as noise or outliers. The underlying assumption for applying DBSCAN to the Labyrinth use case is that the coordinate points sent by the UAV within a given, short enough, time window form one or more dense clusters, and that coordinates clearly deviating from the original path will be easily marked as noise. For this reason, we decide to apply DBSCAN iteratively, over a time window t defined by the final user. For example, for a flight lasting a total of $3'$, with $t = 1'$, we will run three DBSCAN iterations, clustering the points which are recorded within the current window.



Threat Analysis Results(51)

Threat List

Show All... | Search ... | 230 of 230 Threats | Add Threat | Delete Threat | Summary | Print Report

Id	Title	Source	Target	Impact	Likelihood	Risk	Category
T37	Multi protocol attacks	Ground C...	PGN Server	Major	Medium	3	Spoofing
T38	Embedded device updates authenticity and integrity	UAV	UAV	Severe	High	5	Tampering
T39	Embedded device updates authenticity and integrity	Ground C...	Ground C...	Severe	High	5	Tampering
T40	Security breach of the GCS may lead to the UAV attack	Ground C...	Ground C...	Severe	Medium	4	Elevation of
T41	Lack of authorization of control system components	Ground C...	Ground C...	Major	Medium	3	Elevation of
T42	Compromise eNodeB/Femtocell/Microcell	Cell Tower	Ground C...	Moderate	Low	2	Denial of Se
T43	Tansmitting incorrect commands or comand signals	Ground C...	UAV	Severe	High	5	Tampering
T44	Malicious UAS communications attacks	UAV	UAV	Major	Medium	3	Tampering
T45	Malicious UAS communications attacks	Ground C...	Ground C...	Major	Medium	3	Tampering
T46	Malicious UAS communications attacks	Ground C...	Ground C...	Major	Medium	3	Tampering
T47	Malicious UAS communications attacks	PGN Server	PGN Server	Major	Medium	3	Tampering
T48	Malicious UAS communications attacks	Satellite	Satellite	Major	Medium	3	Tampering
T49	Malicious UAS communications attacks	Satellite	Satellite	Major	Medium	3	Tampering
T50	Malicious UAS communications attacks	Satellite	Satellite	Major	Medium	3	Tampering
T51	Malicious UAS communications attacks	Cell Tower	Cell Tower	Major	Medium	3	Tampering
T52	Malicious UAS communications attacks	Cell Tower	Cell Tower	Major	Medium	3	Tampering
T53	Prevent the receiver from detecting unauthenticated signals	Ground C...	PGN Server	Moderate	High	3	Denial of Se
T54	Compromising ADS-B messages	Satellite	Ground C...	Major	Medium	3	Tampering
T55	Compromising ADS-B messages	Satellite	Ground C...	Major	Medium	3	Tampering
T56	Compromising ADS-B messages	Satellite	Ground C...	Major	Medium	3	Tampering
T57	Compromising ADS-B messages	Satellite	IRIDIUM G...	Major	Medium	3	Tampering
T58	Unattended or unauthorized physical access to the UAV	UAV	UAV	Major	Medium	3	Elevation of
T59	Unique intial passwords and keys per device	Bridge Se...	Bridge Se...	Moderate	Medium	2	Spoofing
T60	Access to wireless interface to impersonate parts of the net...	UAV	Ground C...	Negligible	Medium	1	Spoofing
T61	Access to wireless interface to impersonate parts of the net...	PGN Server	Ground C...	Negligible	Medium	1	Spoofing
T62	Access to wireless interface to impersonate parts of the net...	Ground C...	UAV	Negligible	Medium	1	Spoofing
T63	Access to wireless interface to impersonate parts of the net...	Ground C...	PGN Server	Negligible	Medium	1	Spoofing
T64	Wireless APN Floods	PGN Server	Ground C...	Moderate	Low	2	Denial of Se
T65	Wireless APN Floods	Cell Tower	Ground C...	Moderate	Low	2	Denial of Se
T66	Protocol flooding for node bombarding	UAV	Ground C...	Moderate	Medium	2	Denial of Se
T67	Protocol flooding for node bombarding	UTM Server	Ground C...	Moderate	Medium	2	Denial of Se
T68	Protocol flooding for node bombarding	UTM Server	Ground C...	Moderate	Medium	2	Denial of Se
T69	Protocol flooding for node bombarding	PGN Server	Ground C...	Moderate	Medium	2	Denial of Se
T70	Protocol flooding for node bombarding	PGN Server	Ground C...	Moderate	Medium	2	Denial of Se
T71	Protocol flooding for node bombarding	IRIDIUM G...	Ground C...	Moderate	Medium	2	Denial of Se
T72	Protocol flooding for node bombarding	Bridge Se...	Ground C...	Moderate	Medium	2	Denial of Se
T73	Protocol flooding for node bombarding	Bridge Se...	Ground C...	Moderate	Medium	2	Denial of Se

Figure 2: ThreatGet results based on the Labyrinth's high-level model.

By definition, the algorithm requires two other input parameters: ϵ and *min_samples*, respectively the distance threshold used to define the neighborhood of a data point, and the minimum number of points required to form a cluster. While we decide to accept minimum size cluster, setting *min_samples* = 2 for every use case, the estimation of the parameter ϵ requires a more extensive discussion. There are a few approaches to estimating the appropriate value for ϵ in DBSCAN: one could rely on expert knowledge, visual inspection of the data, or trial and error. We decided to estimate the parameter by means of the k-distance plot [11]. This method requires sorting all points by their distance to their k-th nearest neighbor and plotting the distances. The value of k can be determined based on the characteristics of the dataset. In the plot, a significant change in the distance at which points transition from being in dense regions to sparse regions can be seen. This can be typically visualized as a flat region in the distance curve. We then fine-tune the ϵ after a visual inspection. The ϵ parameter estimation is assumed to be performed on anomaly-free, regular flight paths.

Finally, we validate the anomaly detection capabilities of our pipeline. We introduce artificial anomalous points to evaluate the algorithm's output. For each flight, we first select 10 random data points for which we alter latitude and/or longitude — both separated and combined — adding a perturbation in a range from 10^{-6} to 1 decimal degree. We do the same with altitude, adding a perturbation from 10 to 120 meters. We then report the fraction of detection. An example scenario and its results are presented in the following.

5.2 Anomaly Detection Results

By means of the DBSCAN clustering algorithm we aim at isolating anomalous data points in an unsupervised and quasi-real time –– depending on the selected time frame —- fashion. The whole pipeline has been implemented using Python 3[2] and the `pandas`[3] library; the DBSCAN algorithm is provided by the `cluster` module of the `scikit_learn`[4] library.

5.2.1 Flight Data Characterization. For testing our scenario, we profit from a total of 24 recorded flights of different lengths, both

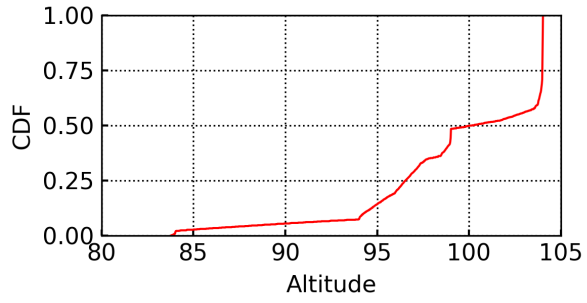
[2] https://www.python.org/about/
[3] https://pandas.pydata.org/
[4] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html

Table 3: Threat Analysis for Location Report.

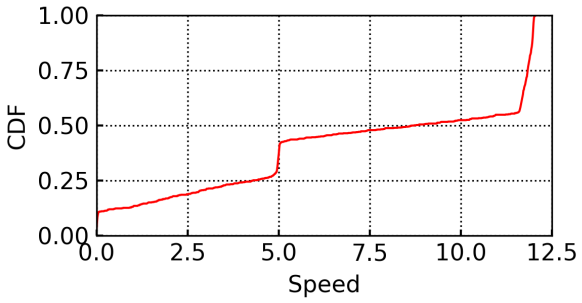
Entity	Type	Source	Destination	Threat	Impact	Rating
Location Report	Flow	UAV	GCS	S	Attacker is taking over the communication channel from the UAV to the GCS.	2 - 4
				T	Change of location information by launching e.g. a man-in-the-middle attack.	2 - 4
				R	-	-
				I	Interception of location reports	1
				D	Interruption of communication channel by DoS attack (e.g. flooding or jamming)	2 - 4
				E	-	-
Location Report	Flow	UAV	UTM	S	Attacker is taking over the communication channel from the UAV to the UTM.	2 - 3
				T	Change of location information by launching e.g. a man-in-the-middle attack.	2 - 4
				R	-	-
				I	Interception of location reports	1
				D	Interruption of communication channel by DoS attack (e.g. flooding or jamming)	2 - 4
				E	-	-
Location Report	Flow	GCS	UTM	S	Attacker is pretending to be a legitimate GCS.	3 - 4
				T	Attacker changing location information sent from the GCS to the UTM.	3 - 4
				R	-	-
				I	Interception of location reports	1
				D	Interruption of communication channel by DoS attack (e.g. flooding or jamming)	2 - 4
				E	-	-

from simulated and real flight tests. During each flight, the UAVs report almost always a number of location points in the order of several hundreds, with a maximum value of 913 measurements and a minimum of 78. The flight length is always reported in minutes and spans from 1h14' to 2'52". Each file contains the spatial and temporal information required to train the algorithm: latitude, longitude, altitude (recorded as `alt`, `alt_rel` or `alt_MSE` depending on the file and referring to the altitude relative to the ground or to the sea level), and timestamp. Note that sometimes multiple timestamps are indicated in the trace. This could depend on where the information exchange is registered or how it was implemented. We have the timestamp provided by the drone indicating when the telemetry was registered, the operator could register the moment when the report reaches the GCS, and the UTM registers when the report arrives to it. We use as a reference the `utm_ts`, but, depending on the specific scenario requirements, also other timestamp indications, e.g., `gcs_timestamp` can be used. We hereby discuss the results for a single flight having a total of 780 points, lasted $24'39''$. For this flight, we analyse the distribution of altitude, speed and points per minute to provide and example of the standard behavior of an UAV. Figure 3 reports the cumulative distribution function of the UAV's altitude (Figure 3(a)), speed (Figure 3(b)) and points per minute (Figure 3(c)). Note that the lowest values in the three measurement categories correspond to the initial and final phases of the flights, namely `start-of-flight` and `eof`, with its specific report messages.

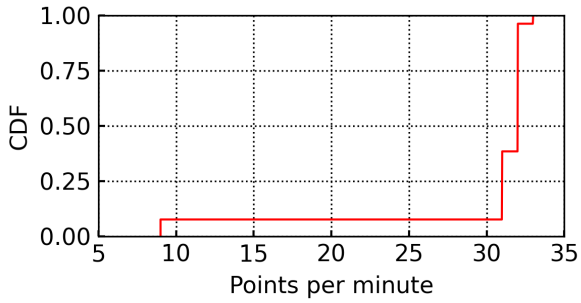
5.2.2 Parameter Estimation. We then proceed with the parameter estimation phase. As mentioned above, our implementation of DBSCAN requires 3 parameters: t, the time window for each clustering iteration, depending on each flight's characteristics; ϵ, the maximum distance from the cluster centroid; and *min_samples*, the minimum number of neighboring samples to form a cluster. Considering the points per minute distribution in Figure 3(c), we set the parameter $t = 1'$, as a smaller time window would lead to an excessively high number of iterations for a very low set of points per iteration. We set *min_samples* = 2, as we decide to accept also clusters of minimum dimension. The evaluation of ϵ requires a closer analysis. We use the k-distance plot method and further tune ϵ by means of trial and error on the perturbation-free flight path. We set k - the number of neighbors – to 9, as 9 is the minimum number of points per minute in our example flight (see Figure 3(c)). Figure 4 reports the results. From the figure, we observe that a reasonable value of epsilon could be estimated at 20 m. This is the distance identified by the flat area from data point 500 on.



(a) Altitude (m) - Avg: 99,62, Max: 104,03, Min: 83,78, Median: 100,17.



(b) Speed (m/s): Avg: 7,51, Max: 12,06, Min: 0, Median: 8,71.



(c) Points per minute - Avg: 29,96153846, Max: 33, Min:9, Median: 32.

Figure 3: Data distribution for altitude, speed, and points per minute.

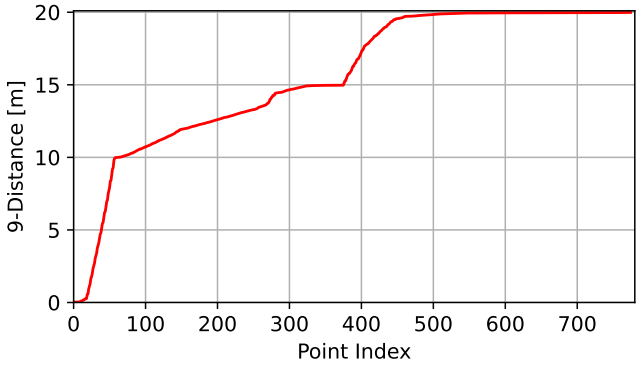
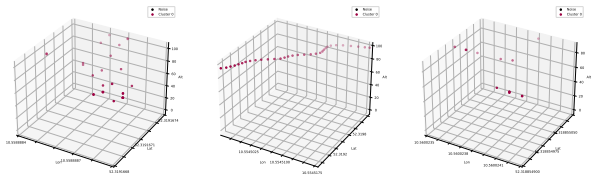


Figure 4: k-Distance plot.

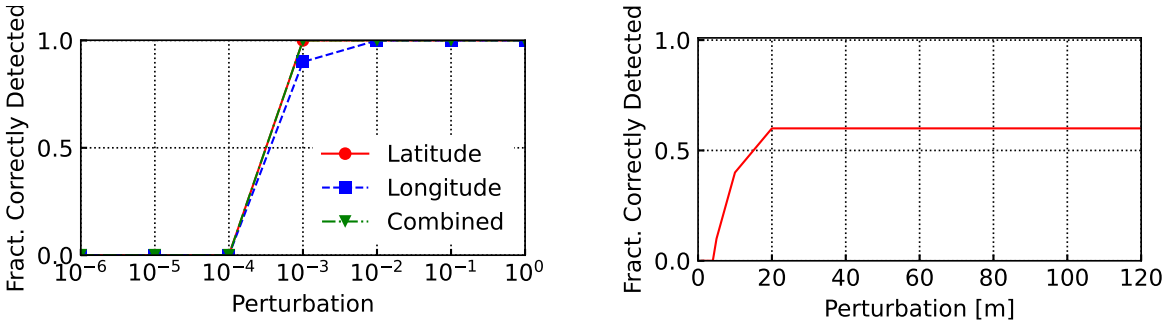


(a) Start of flight (b) Flight (c) End of flight

Figure 5: Example of clustering without noisy points.

5.2.3 Clustering results verification. We initiate the clustering phase with the chosen set of parameters. Figure 5 shows three examples of clustering results on the flight without perturbations: in all the three cases all the points are correctly assigned to a single cluster.

We then verify the algorithm detection capabilities by adding artificial perturbations to 10 randomly chosen samples. The deviations in latitude and longitude vary from 10^{-6} to 1 decimal degrees, both on a single coordinate and combined, while the ones in altitude span from 1 to 120 meters (the limit imposed by law). Figure 6(a) shows the fraction of correctly detected perturbations versus the perturbation's order of magnitude. The algorithm is able to correctly detect all the points showing a variation in latitude, longitude, or in both coordinates from 0.0001 decimal degrees ($11.1m$, concordant to the choice of ϵ). Perturbations in longitude turned out to be slightly harder to detect (0.9 with a perturbation of 0.0001 decimal degrees). Changes in altitude, on the other hand, proved to be harder to detect: Figure 6(b) reports the fraction of correctly classified perturbations in altitude. As we can see, the algorithm detects only the 60% of perturbations, and variations within 5 meters are going undetected. Note that we avoided testing banned or



(a) Latitude and Longitude

(b) Altitude

Figure 6: Algorithm detection capabilities in case of perturbations.

unfeasible values above the limit imposed by law, as those would be detected by Labyrinth's UTM just as the report arrives. The UTM performs lexical and semantic analysis of the reports as they are received. In the case of lexical errors it would deliver an error message; if the error is semantic — values not allowed like altitude over 120 m; values non conformant with respect to the trajectory assigned, like deviations; impossible values considering the capabilities of the drone, etc. — the UTM would trigger an appropriate procedure. Therefore we could say that the UTM also provides a first filter for anomaly detection.

6 CONCLUSION

In this paper we presented a framework for threat identification and anomaly detection in the UAV domain defined during the Labyrinth project. We used ThreatGet as a threat modelling tool to investigate any possible cyber-threat propagating within our Labyrinth system design. We then refined our threat analysis by means of the STRIDE approach. The security analysis is complemented by a lightweight machine learning tool supporting threat detection during flight operations. We focused on detecting significant deviations in the coordinate data points in an unsupervised fashion, given the lack of labelled data points. The process has been considered lightweight enough to be embedded also in the drone; this would allow to detect abnormalities immediately and warn the UTM, increasing its margin of time to apply non-nominal measures and therefore increasing the safety of the airspace. The result validation phase showed advantages and limitations of the framework, proving that it should be perceived as an aiding tool in a broader cybersecurity-oriented scenario, but not used as a standalone detection tool. Nevertheless, Labyrinth introduces different technologies for significant improvements in cybersecurity in the UAV domain and provides a strong foundation for future research and development in cybersecurity in this field.

ACKNOWLEDGMENTS

The research leading to this work has been funded by the Labyrinth Project - European Union's Horizon 2020 research and innovation program under grant agreement No 861696.

REFERENCES

[1] Stuart M Adams and Carol J Friedland. 2011. A survey of unmanned aerial vehicle (UAV) usage for imagery collection in disaster research and management. In *9th international workshop on remote sensing for disaster response*, Vol. 8. 1–8.
[2] Omar Bouhamed, Ouns Bouachir, Moayad Aloqaily, and Ismaeel Al Ridhawi. 2021. Lightweight ids for uav networks: A periodic deep reinforcement learning-based approach. In *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. IEEE, 1032–1037.
[3] Jian Bu, Rui Sun, Hongyang Bai, Rui Xu, Fei Xie, Yucheng Zhang, and Washington Yotto Ochieng. 2017. Integrated method for the UAV navigation sensor anomaly detection. *IET Radar, Sonar & Navigation* 11, 5 (2017), 847–853.
[4] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise.. In *kdd*, Vol. 96. 226–231.
[5] Yongfu He, Yu Peng, Shaojun Wang, Datong Liu, and Philip HW Leong. 2017. A structured sparse subspace learning algorithm for anomaly detection in UAV flight data. *IEEE Transactions on Instrumentation and Measurement* 67, 1 (2017), 90–100.
[6] International Electrotechnical Commission (IEC). 2013-2018. Industrial communication networks - Network and system security - ISO/IEC 62443. (2013-2018).
[7] Seonghoon Jeong, Eunji Park, Kang Uk Seo, Jeong Do Yoo, and Huy Kang Kim. 2021. MUVIDS: False MAVLink Injection Attack Detection in Communication for Unmanned Vehicles. In *Workshop on Automotive and Autonomous Vehicle Security (AutoSec)*, Vol. 2021. 25.
[8] Rafiullah Khan, Kieran McLaughlin, David Laverty, and Sakir Sezer. 2017. STRIDE-based threat modeling for cyber-physical systems. In *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. IEEE, 1–6.
[9] Jun Li, Yifeng Zhou, and Louise Lamont. 2013. Communication architectures and protocols for networking unmanned aerial vehicles. In *2013 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 1415–1420.
[10] J MacQueen. 1965. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symposium on Math., Stat., and Prob.* 281.
[11] Davoud Moulavi, Pablo A Jaskowiak, Ricardo JGB Campello, Arthur Zimek, and Jörg Sander. 2014. Density-based clustering validation. In *Proceedings of the 2014 SIAM international conference on data mining*. SIAM, 839–847.
[12] G Panice, Salvatore Luongo, Gabriella Gigante, Domenico Pascarella, Carlo Di Benedetto, Angela Vozella, and Antonio Pescapè. 2017. A SVM-based detection approach for GPS spoofing attacks to UAV. In *2017 23rd International Conference on Automation and Computing (ICAC)*. IEEE, 1–11.
[13] Tarek Rakha and Alice Gorodetsky. 2018. Review of Unmanned Aerial System (UAS) applications in the built environment: Towards automated building inspection procedures using drones. *Automation in Construction* 93 (2018), 252–264.
[14] Catur Aries Rokhmana. 2015. The potential of UAV-based remote sensing for supporting precision agriculture in Indonesia. *Procedia Environmental Sciences* 24 (2015), 245–253.
[15] Mohammad Sajid, Himanshu Mittal, Shreya Pare, and Mukesh Prasad. 2022. Routing and scheduling optimization for UAV assisted delivery system: A hybrid approach. *Applied Soft Computing* 126 (2022), 109225.
[16] Christoph Schmittner, Sebastian Chlup, Andreas Fellner, Georg Macher, and Eugen Brenner. 2020. ThreatGet: Threat modeling based approach for automated and connected vehicle systems. In *AmE 2020-Automotive meets Electronics; 11th GMM-Symposium*. VDE, 1–3.
[17] Lei Shen. 2014. The NIST cybersecurity framework: Overview and potential impacts. *Scitech Lawyer* 10, 4 (2014), 16.

[18] Daniel P Shepard, Jahshan A Bhatti, Todd E Humphreys, and Aaron A Fansler. 2012. Evaluation of smart grid and civilian UAV vulnerability to GPS spoofing attacks. In *Proceedings of the 25th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2012)*. 3591–3605.

[19] Rui Sun, Qi Cheng, Guanyu Wang, and Washington Yotto Ochieng. 2017. A novel online data-driven algorithm for detecting UAV navigation sensor faults. *Sensors* 17, 10 (2017), 2243.

[20] Nils Ole Tippenhauer, Christina Pöpper, Kasper Bonne Rasmussen, and Srdjan Capkun. 2011. On the requirements for successful GPS spoofing attacks. In *Proceedings of the 18th ACM conference on Computer and communications security*. 75–86.

[21] Benkuan Wang, Zeyang Wang, Liansheng Liu, Datong Liu, and Xiyuan Peng. 2019. Data-driven anomaly detection for UAV sensor data based on deep learning prediction model. In *2019 Prognostics and System Health Management Conference (PHM-Paris)*. IEEE, 286–290.

[22] Chunhua Zhang and John M Kovacs. 2012. The application of small unmanned aerial systems for precision agriculture: a review. *Precision agriculture* 13 (2012), 693–712.