EINDHOVEN UNIVERSITY OF TECHNOLOGY

UNIVERSITÀ DEGLI STUDI DI PADOVA

Department of Mathematics and Computer Science
Security Research Group

Energy Depletion Attacks on Battery-powered IoT Devices

Master Thesis

Kourosh Marjouei

Graduation Supervisors:
Dr. S. Sciancalepore, Technische Universiteit Eindhoven (TU/e), Netherlands
Dr. A. Brighente, University of Padova, Italy

Eindhoven, July 2025

Abstract

Unmanned aerial vehicles (UAVs), commonly known as drones, are rapidly gaining popularity for applications in delivery services, emergency rescues, and military operations. Drones are often powered by energy-constrained batteries, making them vulnerable to energy depletion attacks. These attacks aim to exhaust UAV batteries by maliciously targeting existing wireless communication channels or directly contacting UAV physical components. Meanwhile, Physical Sensor Attacks (PSAs) and False Actuation Injection (FAI) attacks are two major physical-layer attacks against UAVs that manipulate the victim's environment by remotely injecting or jamming signals to form an implicit control channel on the victim's sensors or actuators without establishing any physical interactions or network connections with the victim UAV. Indirectly, such attacks can cause energy depletion on drones. However, to the best of our knowledge, no previous research investigated the extent to which PSAs and FAI attacks can be used as energy depletion attacks. This thesis addresses the main research question: *Can existing PSAs and FAI attacks be used for conducting the energy depletion attack?* To answer the question, we propose a probabilistic attack framework algorithm for exhausting the battery of an UAV by considering PSAs and the FAI attack as attack vectors, which are prioritized based on their direct or indirect impacts on rotating the servo motors of the target UAV. We experimentally evaluate the performance of our proposed attacks through simulations in Ardupilot software-in-the-loop (SITL), by modifying the related simulation parameters of a target drone. We performed two categories of attack experiments, namely *1-min* and *full-battery*, and evaluated the impact of such attacks for 1-minute attack duration and until the battery exhaustion, respectively. Based on our experimental results, the most successful attack type is the FAI attack with its maximum attack intensity due to its direct impact on servo speed and rotation, showing over 16% increase in battery depletion for 1-min attack experiments and over 200 seconds (s) decrease in average exhaustion time for full-time attack experiments compared to the observed results under no attack.

Keywords: Energy-constrained IoT Devices, UAVs, Drones, Physical Sensor Attacks, Energy Depletion Attacks, Battery Exhaustion, ArduPilot SITL.

Preface

This report presents the findings of my master's thesis, which was completed as part of the requirements for the Master's Degree in Computer Science at the University of Padova, Italy. The research was conducted at the Security Research Group at Eindhoven University of Technology (TU/e) during my exchange mobility in the Netherlands. I have thoroughly enjoyed the time I spent studying and the side activities in both countries. Furthermore, I have gained a lot of new knowledge that I will apply to the best of my ability in my future career. During my master, several people have been an important part to which I want to express my appreciation to.

First, I would like to express my deepest gratitude to my graduation supervisors, Savio Sciancalepore and Alessandro Brighente, their expertise and feedback played a crucial role in shaping my research. I greatly appreciated your patience and time during and outside the periodic meetings, which helped me greatly develop my analytical skills and the ability to present new insights on the analysis.

Second, I want to thank my family for their unconditional love, support, and motivation during my entire life, and I am very grateful for that.

Finally, both my "UniPD" and "TU/e" friends, for the friendship we have created during my master's program. I could always rely on them for discussion during courses or regarding my master's thesis, and after-class activities.

Enjoy reading my thesis.

Kourosh Marjouei
Eindhoven, July 2025

Contents

Contents vii

List of Figures ix

List of Tables xi

Listings xiii

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Research Question . . . 2
- 1.3 Report Outline . . . 2

2 Background 3
- 2.1 General Architecture of UAVs . . . 3

3 Literature Review 5

4 System and Adversary Model 9
- 4.1 System Model . . . 9
- 4.2 Adversary Model . . . 10

5 Attack Methodology 13
- 5.1 Attack Requirements . . . 13
- 5.2 Rationale of the Attack . . . 14
- 5.3 Attack Framework Algorithm . . . 14
- 5.4 Attack Strategy . . . 16

6 Experimental Results 19
- 6.1 Experimental Setup . . . 19
- 6.2 Experiment Settings . . . 20
- 6.3 Experiment Plan . . . 23
  - 6.3.1 Attacks Implementation . . . 24
- 6.4 Performance Metrics . . . 25
- 6.5 Simulation Results . . . 26
  - 6.5.1 One-minute Attack Experimental Results . . . 26
  - 6.5.2 Full-battery Attack Experimental Results . . . 33

7 Conclusions 37
- 7.1 Conclusion . . . 37
- 7.2 Limitations and future work . . . 38

Bibliography 39

List of Figures

2.1 Block diagram of an UAV system (adapted from [41]). 3
2.2 The mechanism of how attitude control works on an UAV. [19] 4

4.1 Scenario assumed in this work. 9

6.1 Impact of the FAI attack on both the total average energy consumption of a simulated quadcopter and its corresponding average battery depletion for a 1-min attack experiment. 27
6.2 Impact of GPS spoofing attack on both total average energy consumption of a simulated quadcopter and its corresponding average battery depletion for a 1-min attack experiment. 28
6.3 Impact of optical flow spoofing attack on both total average energy consumption of a simulated quadcopter and its corresponding average battery depletion for 1-min attack experiment. 29
6.4 Impact of EMI attack on SPI channels on both total average energy consumption of a simulated quadcopter and its corresponding average battery depletion for 1-min attack experiment. 29
6.5 Impact of spoofing accelerometer sensor values (Xacc, Yacc, Zacc) on both total average energy consumption and its corresponding average battery depletion percentage. 31
6.6 Impact of spoofing gyroscope sensor values (Xgyro, Ygyro, Zgyro) on total average energy consumption and its corresponding average battery depletion percentage. . 32
6.7 Impact of coupling attacks on both total average energy consumption of a simulated quadcopter and its corresponding average battery depletion for 1-min attack experiment. 33
6.8 Overall comparison of attack types tested for full-battery attack experiments. . . . 34
6.9 A detailed comparison of weak attacks tested for full-battery attack experiments. . 34
6.10 Impact of weak coupled attacks on average exhaustion time of a simulated quadcopter for full-battery experiment. 35
6.11 Impact of strong coupled attacks on average exhaustion time of a simulated quadcopter for full-battery experiment. 35

List of Tables

3.1 An overview of the literature review on EDAs and the existing research gap in conducting both PSAs and FAI attacks to deplete energy of a target UAV. 6
3.2 Six different types of PSAs with two different levels of adversary capabilities [21]. . 7
3.3 An example of prioritized physical-layer attacks based on their direct or indirect impacts on rotating the servo motors of an UAV. 8

4.1 Equipment list for carrying out different PSAs and FAI attacks. 11

5.1 A list of attacker's requirements and their corresponding motivations. 13
5.2 Attacker's requirements and their corresponding attack strategy. 16

6.1 Our general experiment settings before running attack experiments in ArduCopter SITL. 20
6.2 The related SITL parameters implemented by [21], and we also use them in this work to add an optical flow sensor in ArduCopter SITL before conducting optical flow spoofing. 21
6.3 The configuration of SITL parameters identified and modified in this work to simulate our attack experiments, adding both weather conditions and motor thrust randomness to provide the statistical variability. 22
6.4 A comparison of different attack types simulated in this work for both 1-min and full-battery experiments. 23
6.5 The classification of SITL parameters and MAVLink commands used for simulating PSAs and FAI attacks in ArduPilot SITL. 25
6.6 Performance metrics used in our experiments. 25

7.1 Research sub-questions and their answers. 37

Listings

Chapter 1

Introduction

This chapter is an introduction to the report. First, the motivation for conducting this research is given. After the motivation is clear, the problem with the current situation is more clearly defined, and the main research question is introduced. Lastly, we will provide a report outline stating the content of each chapter.

1.1 Motivation

With the rapid development of science and technology, UAVs, also known as drones, are playing an increasingly important role in our society [63]. In fact, UAVs respond to the requirements of various missions over considerable application areas such as agriculture, media coverage, emergency services, healthcare, object detection, tracking, surveillance, and data collection [51]. Drones rely heavily on sensors such as gyroscopes, accelerometers, barometers, and optical flow sensors, as well as Global Navigation Satellite System (GNSS) receiver, so as to navigate their environment, perform their given task, and deal with various unforeseen conditions for stable flight. In addition, most drones are equipped with energy-constrained batteries, making energy a critical resource during the mission of drone.

Despite the increasing popularity and their many advantages, security attacks targeting UAVs have become a serious concern in recent years [21]. One of these attacks falls into the category of Energy Depletion Attacks (EDAs), which exploit the limited energy and battery lifetime in UAVs. To be more specific, there are two classes of energy exhaustion attacks on UAVs. The first one influences wireless communication channels (e.g., GPS signals or control data transfer channel) without physical contact with the target. The second class of EDAs attacks is based on physical contact with the target UAV. For example, the attacker can exhaust the drone's energy by connecting a malicious USB device to the drone's USB interface. Replacing the battery for a faulty or degraded battery is another example of this class of EDAs. The ultimate attack goal for both forms of EDAs is to exhaust the battery of a target UAV, disrupting its mission and eventually a crash [9], [51].

Another major category of attacks against UAVs is known as physical-layer attacks, which can be divided into PSAs and FAI attacks [21], [8]. While PSAs are attacks that target sensor measurements and GNSS signals, the FAI attack focuses on manipulating the actuation signals (i.e., PWM signals) that control the speed and rotation of a servo motor of a victim UAV. Both PSAs and FAI attacks manipulate the victim's environment by remotely injecting or jamming signals to form an implicit control channel on the victim's sensors or actuators without establishing any physical interactions or network connections with the victim UAV. The main attack goals are mission failure, unstable attitude/position, or a physical crash.

Although it is quite well known that such attacks could indirectly be used to deplete the energy of the drone, to the best of our knowledge, no systematic investigation of their effectiveness in such regard exists, and not on drones.

1.2 Research Question

Based on the above motivation, we can see that there is a research gap in the existing literature. This thesis aims to address this gap by evaluating for the first time the impact of PSAs and FAI attacks on the energy consumption of a target energy-constrained IoT device (e.g., UAVs) in a simulation platform. Therefore, the central research question of this study is:

Can existing PSAs and FAI attacks be used for conducting the energy depletion attack?

To address this question, the research will focus on the following sub-questions:

Q1. What specific sensor/actuator packets can affect and maximize the energy consumption in Low Power Wireless (LPW) devices, and give a ranking to each one of these sensors?

Q2. How can the adversary utilize physical-layer attacks to conduct energy depletion attacks?

Q3. What are the attacker's requirements and their strategies for successfully conducting the battery exhaustion attack?

Q4. What effect does this attack have on total energy consumption and battery depletion percentage within 1 minute?

Q5. What effect does this attack have on the average exhaustion time?

To answer these questions, we proposed a probabilistic attack framework algorithm for exhausting the battery of an UAV, using a list of attack vectors (including both PSAs and FAI attacks) which are prioritized based on their direct or indirect impacts on rotating the servo motors of a target drone. This algorithm then returns a list of successful attack vectors with the highest probability, resulting in the exhaustion of the drone's battery, thereby forcing the drone to land. We also perform two categories of experiments to evaluate our analysis in a simulation platform. The first category focuses on investigating the impact of physical-layer attacks on a simulated quadcopter energy consumption, and its corresponding battery depletion percentage for 1 minute attack duration. Another category is full-battery attack experiments, in which we analyze the results of average exhaustion time for different attack types by extending the duration of experiments until battery exhaustion. Evaluation of 1-min and full-battery attack experiments demonstrated, respectively, over 16% increase in average battery depletion, and 200 s decrease in average exhaustion time when the attacker utilized the FAI attack with its maximum attack intensity compared to the results observed from the simulated drone, which is flying under no attack. However, the results obtained from testing other attacks (i.e., PSAs) showed more limited impact on the given performance metrics.

1.3 Report Outline

This report is organized into several key sections. In Chapter 1, the objectives and scope of the research are outlined. We introduce different components of a generic UAV system and their functionalities in Chapter 2. Moving on to Chapter 3, the existing literature and relevant research are reviewed to provide the current research gap, which is addressed in this study. Chapter 4 describes the capabilities of an adversary and the environment in which he will attack. In Chapter 5, the attack framework methodology for conducting the research is explained. In addition, both the requirements utilized for a successful attack and their corresponding attack strategy are outlined. An explanation of how we acquired our data and the experimental setup used is described at the beginning of Chapter 6. This chapter also presents the results and analysis, providing an extensive experimental evaluation of the data and findings from the experiments. Finally, Chapter 7 concludes the report, summarizing the key points and suggesting future research directions.

Chapter 2

Background

In this section, we provide background information for a general system overview of an UAV, introducing all sensors that are used later on as part of the energy depletion attack.

2.1 General Architecture of UAVs

An UAV system generally consists of three components, namely the UAV, the Ground Control Station (GCS), and the Communication Link (CL) [26]. The relationship between these three components is shown in Figure 2.1.

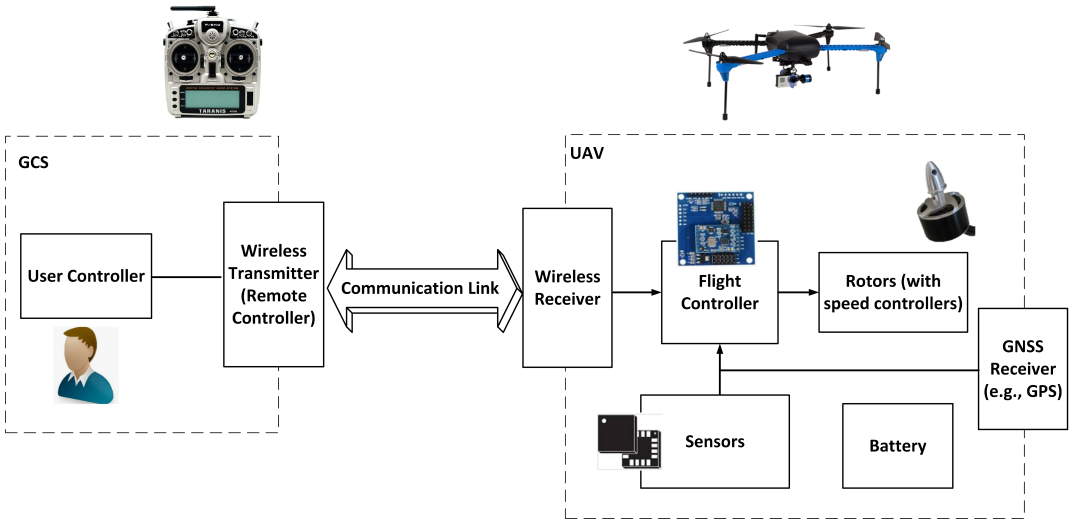


Figure 2.1: Block diagram of an UAV system (adapted from [41]).

The inner hardware architecture of an UAV includes: a flight controller, a rechargeable battery, actuators (i.e., multiple rotors), a set of (possibly redundant) homogeneous and heterogeneous sensors, a GNSS receiver, and a wireless receiver.

The flight controller serves as a control unit of the UAV, and takes two inputs: one from the GCS wireless transmitter through the UAV wireless receiver, and the other from different types of sensors and the GNSS receiver. The former and latter inputs are called the communication channel and sensing channel, respectively. Based on these two inputs from two channels, the flight controller determines the attitude and position of the drone, including the speed of rotors [41].

In order to provide the power supply for the whole UAV, the drone leverages energy-constrained batteries that are rechargeable. Another main components of an UAV are actuators that produce

the appropriate actuation needed for the UAV during the flight mission, thus ensuring high stability. A specific type of widely used actuators in UAVs is called Pulse Width Modulation (PWM)-based actuators that consist of the brushed (DC) motors, and the propellers which are controlled by Electronic Speed Controllers (ESCs). Note that an ESC regulates the rotation of the motors to achieve the desired rotational speed that is controlled by PWM [8].

UAVs use a wide variety of sensors to fly safely: 1) Inertial Measurement Unit (IMU) sensors that include accelerometers and gyroscopes, each one measuring a particular UAV physical quantity, namely the UAV's attitude (roll, pitch, and yaw angles), and angular velocity, respectively. 2) Magnetometers that provide additional geographical direction for UAVs using the magnetic field. 3) Position sensors, including a GNSS receiver and an optical flow sensor that determine the position of the UAV, using data from multiple GNSS constellations, and the background features of the captured image, respectively. Note that the GNSS receiver can also be used to measure the UAV's attitude (yaw angle) [21].

The measured quantities from sensors are then processed, transferred, and interpreted as digitized values for the drone's attitude control. Using these retrieved values, the control unit calculates the difference between the current and desired attitude for a stable flight and then determines appropriate commands to be sent to the rotors. During this process, the values of various sensors can be combined in a complementary way to reduce calculation errors. This is often referred to as sensor fusion. For instance, the Extended Kalman Filter (EKF) is a well-known sensor fusion method. It is used to precisely estimate a drone's current attitude by combining the values of the IMU's sensors. By repeating these steps, the drone can maintain its posture, thereby resulting in a stable flight [19] (See Fig. 2.2).

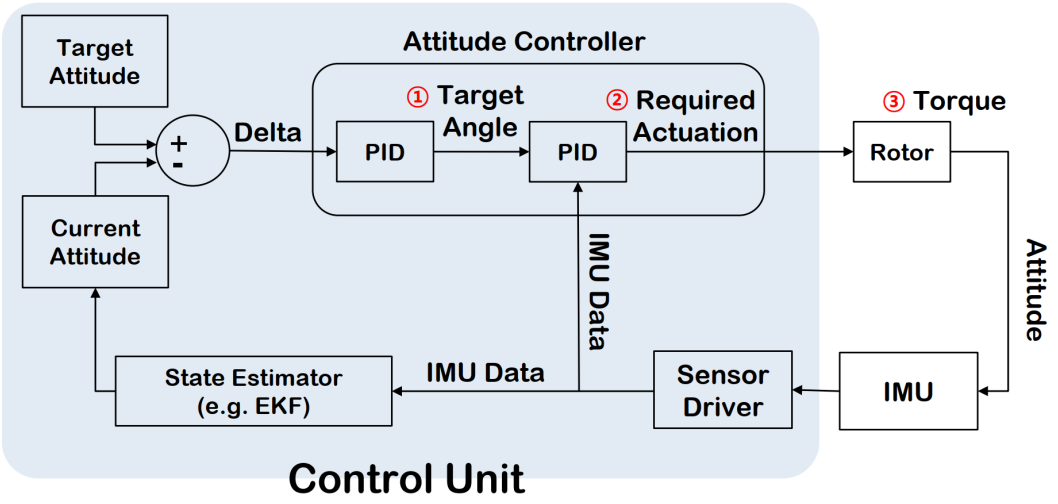


Figure 2.2: The mechanism of how attitude control works on an UAV. [19]

Note that many security attacks targeting UAVs rely on disrupting their sensing channel and the actuation signal. They fall into two categories, namely "Physical Sensor attacks" and "False Actuation Injection" attacks. In the next chapter, we discuss in more detail different attack types classified into these two categories based on their attack target.

Chapter 3

Literature Review

In this chapter, we first review the existing works on Energy Depletion Attacks (EDAs) in Internet of Things (IoT) networks, specifically those attacks targeting Low Power Wireless (LPW) networks and UAVs. Next, two other categories of attacks affecting UAVs, namely PSAs and the FAI attacks, are analyzed. By comparing all different attack types depleting the energy of the UAV, we show the current research gap in the literature. In fact, to the best of our knowledge, this project for the first time addresses the research gap of conducting energy depletion attacks on energy-constrained IoT devices by utilizing both PSAs and the FAI attacks without any physical interactions or established network connections with the target UAV. The Table 3.1 shows the existing research gap addressed by this project.

According to [28], EDAs in LPW networks can be classified in four protocol layers (see [28] for a comprehensive overview). As can be seen from Table 3.1, most works on EDAs focused on LPW networks. To be more specific, some types of jamming attacks are generally considered a form of energy depletion for LPW networks [29]. Specifically, deceptive and reactive jamming attacks lead a target to receive and process a standard-compliant packet [5], emerging as a form of energy depletion. However, it is necessary for these types of jamming attacks to have a previous knowledge of IEEE 802.15.4 specifications for both physical and Medium Access Control (MAC) layers (e.g., modulation scheme, and channel access). To launch an effective and stealthy EDA in each layer, the attacker can exploit the found vulnerabilities or use a suitable scenario in available tools (e.g., [60, 36] to attack. In other words, for all different protocol layers in LPW networks, the adversary is required to establish a successful network connection to the victim LPW device(s) before conducting the attack.

Overall, the authors in [9, 51] introduced two main classes of EDAs targeting UAVs: The first one includes wireless channel attacks. This class influences existing wireless communication channels (i.e, wireless control data channel, GPS channel) and does not assume any physical contact with the drone. On the other hand, the second class of attacks targets UAV physical components and presumes the physical contact of the attacker with the drone. This second class of EDA attack types is successful when the adversary has full physical access to the drone in a standby mode. For example, the attacker can exhaust the drone's energy by connecting a malicious USB device to the drone's USB interface. Replacing the battery for a faulty or degraded battery is another example of this class of EDAs (see [9] for more examples).

Focusing on the first class of EDAs targeting wireless control data channels in UAVs, [51] highlighted that the adversary can indirectly deplete the UAV battery by sending continuous wireless bogus communication packets through the ZigBee network channel, causing the UAV to run part of an energy-consuming authentication protocol for analyzing every request. This type of attack falls into the category of "Denial of Service (DoS)" attacks targeting UAVs introduced in [9, 51, 23]. Other types of UAV DoS attacks are jamming attacks that generate electromagnetic noise, causing high error rates at the UAV, thereby increasing the number of re-transmissions. As a result, the UAV may be forced to raise transmission power, which affects battery life. Also, the authors in [10] identified the software vulnerabilities in the MAVLink Protocol, which is used

References	Attack Type	Attack Point	Focus on UAVs	Lack of Interaction
[28], [29], [59], [61] [27], [5], [3], [31], [30], [57], [16], [12], [38]	EDAs	Protocol Layers	×	×
[51], [9]	EDAs	Physical Components	✓	×
[10], [51],[23], [56]	EDAs (e.g., DoS Attacks)	Control Data Channel (e.g., ZigBee Channel)	✓	×
[51], [9], [23]	EDAs (i.e., GPS Jamming, GPS Spoofing)	GPS Data Channel	✓	✓
[21], [7], [19], [20], [33], [41], [52], [53], [34], [62], [58]	PSAs	Sensor Output	✓	✓
[8], [37]	FAI	Actuation Signal	✓	✓

Table 3.1: An overview of the literature review on EDAs and the existing research gap in conducting both PSAs and FAI attacks to deplete energy of a target UAV.

for the bidirectional communication between a drone and a GCS. The attacker can exploit the found vulnerabilities and replay some modified data captured in the past to perform a form of DoS attack. To sum up, these DoS attacks focused on the wireless network vulnerabilities found in the application layer of a general UAV model. So, they established a successful remote connection to the victim UAV before conducting their malicious attacks.

Regarding another first class of EDAs targeting the GPS channel, the authors in [51], [9], [23] proposed utilizing GPS jamming and GPS spoofing to indirectly deplete the energy of the victim UAV. Yet, no experiments were done to show how GPS jamming or spoofing can be effective in depleting the energy of the target UAV.

At the same time, sensors and DC-based servo motors have already been shown to be vulnerable to spoofing in UAVs [7, 14, 37]. In fact, both PSAs and FAI attacks against UAVs have become serious concerns due to their prevalence and potential physical threats [21, 8]. Their main attack goal is to disrupt both the UAV's sensors and PWM-controlled actuators (e.g., DC servo motors), resulting in a mission failure, unstable attitude or position control, and physical crash. To be more specific, the adversary manipulates the victim's environment by remotely injecting/jamming signals to form an implicit control channel on the victim's sensors or actuators. Note that according to [21], PSAs consist of all sensor attacks and GPS spoofing/jamming attacks.

According to Table 3.2 presented in [21], two distinct levels of PSAs can be found in the literature based on their capabilities: i) the attacks that have been proven by conducting real-world experiments so that the attacks demonstrate "*Proven Impacts*" on a target UAV ii) The attacks that possess the most powerful adversary's capabilities so that they can have "*Claimed*

Impacts" on the UAV by overcoming hardware and software limitations.

References	Attack Type	Attacker Capabilities	
		Proven Impact	**Claimed Impact**
[41], [52], [58], [53], [20]	Acoustic Noise Injection	One IMU	All IMUs
[19]	EMI Injection on SPI/I2C	Partial corruption of all sensors	Full corruption of all sensors
[33]	EMI Injection on Magnetometer	One mag sensor	All mag sensors
[34]	GNSS Spoofing	GPS signals	GNSS signals
[34]	GNSS Jamming	Intermittently jam	Fully jam
[7]	Optical Flow Spoofing	Partial field of view	Entire field of view

Table 3.2: Six different types of PSAs with two different levels of adversary capabilities [21].

Considering examples of PSAs, acoustic noise injection attacks demonstrated in the papers [41], [52], [58], [53], [20] inject sound noise at a resonance frequency of a target sensor (i.e., accelerometer or gyroscope) in one of the IMUs on an UAV by activating a microphone or a long-range acoustic device. However, the attacker can impact all IMU sensors by injecting sound at multiple frequencies at the same time because each IMU may have a different resonance frequency. This type of impact is a claimed impact since the attacker needs to overcome the challenges of injecting sound at multiple frequencies.

Another example of PSAs against UAVs is Electromagnetic interference (EMI) signal injection. The works in [33] and [19] leverage EMI to disturb magnetometers and communication channels between IMU sensors and the flight control unit (i.e., SPI/I2C buses), respectively. Based on an adversary's capabilities (as shown in Table 3.2), the EMI attack on magnetometers and SPI/I2C buses can impact a victim UAV in two different ways.

Moreover, an UAV is also equipped with a GNSS receiver [1] which plays an important role in measuring both the UAV's attitude (yaw angles) and its position. However, GPS receivers are vulnerable to spoofing and jamming attacks due to two main reasons [21]: Firstly, the strength of legitimate GPS signals is extremely weak (-125 dBm to -130 dBm in open space). Secondly, civil GPS receivers do not leverage authentication methods compared to military ones [63]. Attackers deceive GPS receivers by either spoofing fake signals that are stronger than legitimate ones or by jamming legitimate signals [62], [50], [34]. Such attacks may lead to unstable position control and a physical crash into an obstacle. In order to reduce the influence of drift without GPS, some UAVs are equipped with optical flow sensors. It means an UAV can distinguish whether it is drifting by comparing the pixel of the adjacent picture. Meanwhile, the authors in [7] show that optical flow sensors in UAVs can be vulnerable to spoofing by injecting images on floors via laser beams, causing the UAV's optical flow sensor to calculate fake motions. The proposed optical flow spoofing attack results in unstable position control and potentially causes a crash into an obstacle.

Finally, the authors in [37] focused on deploying an attack that manipulates the rotation of a servo by changing the duration of PWM signals, impacting the angular position of the servo. Also, [8] conducted an Intentional EMI (IEMI) attack targeting PWM-controlled actuators. The main attack goal is to compromise the servo motors of a fixed-wing UAV by altering PWM signals through EM coupling, making the UAV uncontrollable.

By comparing the current research works on EDAs targeting UAVs with available PSAs and

[1] GNSS receivers use data from multiple satellite constellations such as GPS, Galileo, GLONASS, BeiDou, and SBAS. GPS is the typical constellation used across all UAV platforms and manufacturers [34]

the FAI attacks against UAVs (as presented in Table 3.1), this work addresses a research gap in investigating how the PSAs and the FAI attacks can be used to exhaust the energy of UAVs.

Additionally, the authors in [2], [13] have carried out theoretical and empirical power consumption models for UAVs so as to identify the factors that have an influence on an UAV's power consumption. In general, apart from environmental factors (i.e., weather conditions) such as wind speed, air pressure, and the temperature, the UAV power consumption consists of two main parts [13]: i) communication-related power consumption due to the broadcast Wi-Fi or GPS signals, LED indicators, and internal processing (e.g., EKF switching). ii) flight power consumption, which depends on different flight scenarios (e.g., hovering, horizontal or vertical flight, vertical ascend/descend), which can directly/indirectly control servo motors for rotating propellers.

According to [13], the flight power consumption is usually much more significant than communication-related power expenditure. Thus, it is important to understand how the UAV power consumption varies according to different PSAs and FAI attacks. We come up with Table 3.3, which prioritizes these attacks based on the relationship between an UAV variable (controlled by the sensor/actuation data or by a GPS receiver) with the UAV power consumption. For example, the FAI attack, which has a direct effect on the activation of UAV's motors, is shown on the top of Table 3.3.

Attack Type	**Attack Target**	**Impacted Variable**	**Reasons for Depletion**
FAI	Servo motor	PWM signals	Directly control the rotation angle Directly change the motor speed
GPS Spoofing	GPS receiver	GPS signals	Change the navigation Change the velocity Indirectly control the motor spin
Optical flow Spoofing	Optical flow sensor	Image pixels Background features	Change the navigation Indirectly control motor spin
EMI injection on SPI/I2C	Communication channel (e.g., SPI/I2C buses)	All sensor values	Change Attitude/Position Activate EKF Switching Indirectly change rotor speed
EMI injection on Mag	Magnetometers	Yaw (Heading) measured by Magnetic field	Change Attitude Activate EKF Switching Indirectly change the rotor speed
Acoustic Noise Injection	IMUs (Gyroscopes & Accelerometers)	Angular rates Linear acceleration	Change Attitude/Position Activate EKF Switching Indirectly change rotor speed

Table 3.3: An example of prioritized physical-layer attacks based on their direct or indirect impacts on rotating the servo motors of an UAV.

Chapter 4

System and Adversary Model

4.1 System Model

The scenario considered in this work is depicted in Fig. 4.1

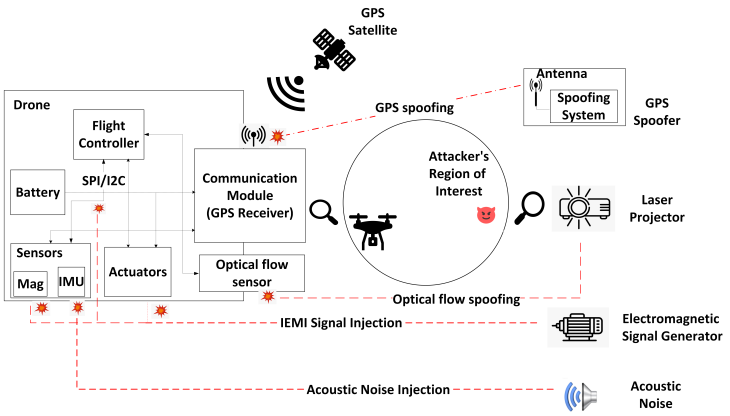


Figure 4.1: Scenario assumed in this work.

We consider two main entities: i) the drone and ii) the adversary. We consider a generic drone with limited energy and limited battery lifetime, namely d_n, flying in an area to achieve the intended mission. It should be noted that d_n might be either remotely piloted, by either a user or a GCS, or (semi)-autonomous, performing a series of autonomous tasks. For convenience, independently of remote control, we will refer to the GCS as the entity responsible for the drone movement. The general components of the drone can be seen in Figure 4.1. In line with modern capabilities and features, we assume d_n features various sensors, including accelerometers and gyroscopes inside of IMU, magnetometers, and barometers.

In addition, we assume that the drone is equipped with a GNSS receiver, e.g., a GPS receiver, so as to be able to compute its own actual location from GPS satellites, in terms of latitude, longitude, and altitude. It should be noted that the drone with a generic GNSS receiver uses data from multiple satellite constellations such as GPS, Galileo, GLONASS, BeiDou, and SBAS. Further, we consider a drone featuring an optical flow sensor, used for lateral stabilization and navigation in non-GPS environments. Specifically, the GPS receiver of a drone is not working if the drone is flying in specific flight modes (e.g., stabilize, Guided_NoGPS) or in non-GPS environments (e.g., indoor environments, military areas, dense forests). In these flight scenarios and environments, the drone relies only on non-GPS equipment such as inertial sensors (IMUs) and vision sensors (e.g., Optical flow sensors) for navigation. Also, we assume that the target drone system fully trusts the data provided by all sensors and actuators. This assumption is

reasonable since most sensors do not provide any built-in mechanism for distinguishing abnormal and normal readings [41].

4.2 Adversary Model

Overall, the objective of the adversary is to capture the drone without crashing it. Satisfying the condition that the drone must not crash during the attack can provide the adversary with access to other resources of the target drone, such as memory. To achieve the attack goal, the attacker conducts stealthy energy depletion attacks on the target drone as quickly and effectively as possible by utilizing both PSAs and the FAI attacks without any physical interactions or established network connections with the drone.

It is important to note that the attacker can use as attack vectors all physical-layer attacks as given in Table 3.3. Noted that we exclude GPS jamming attacks due to two main considerations: 1) Jamming attacks are generally not stealthy, and thus, easy to detect, and 2) these attacks block the perceived data instead of manipulating the GPS signals. So, an adversary has less flexibility to capture the drone in a controlled way, causing the UAV to crash immediately.At the same time, the attacker must also keep the drone in a specific attack zone or Region of Interest (ROI) so as to ensure that the target drone is consistently under the attack, thus depleting its energy quicker. Eventually, the battery of the target drone is exhausted, forcing it to land.

Regarding capturing and eventually forcing the target drone to land, our attack model is a more comprehensive and robust approach compared to a smart GPS spoofing attack (e.g., spoofing the location of a no-fly zone, so as to let the drone immediately land). Our proposed attack model covers not only GPS spoofing, but also other physical-layer attacks. So, we can select and carry out other attack types to attack the drone in case the drone is immune to GPS spoofing.

It should also be noted that, before conducting the attack, the attacker should preliminarily identify the model of the target drone. This preliminary step of the attack can improve the adversary's knowledge of i) the target drone types (e.g., quadcopter, fixed-wing), ii) drone brands (e.g., DJI Phantom 4, Parrot Bebop 2), iii) available sensors on the drone, and iv) the drone movement (e.g., if the drone is increasing its altitude, moving forward, backward). Specifically, if the attacker detects the target drone type and brand (using detection methods such as [17], [35], [22]), it can dynamically prioritize and select an optimal energy depletion attack from the available PSAs and the FAI attacks, thereby increasing the likelihood of achieving the attack goals. In case the attacker is not successful to identify the drone model, the attacker conducts a default list of attack vectors, which are prioritized based on their energy consumption level for a specific well-known drone model (see Sec. 5.3). We also assume that the adversary has knowledge that the drone is intended to consume more power consumption during some specific flight statuses (See [13] for more details). So, the adversary can utilize this knowledge for building attack scenarios in a controlled way, causing the drone to not only deplete more energy, but also stay in the attacker's ROI.

In this work, we also assume that the adversary has the minimum capability to control the environment by 1) manipulating multiple sensors readings inside of an IMU at the same time, 2) fully corrupting all sensor data on SPI/I2C buses, 3) spoofing all mag sensors simultaneously 4) spoofing GPS receiver signals, 5) altering the entire field of view captured by an optical flow sensor, and 6) manipulating the rotation of one servo motor of the target drone in a given ROI, using multiple equipment located in the attack zone. This assumption solves the dependency issue between the type of attack vector and the attacker's ROI. Without considering this assumption, the adversary's capability is limited to launching a subclass of attack types. For example, if the attacker's ROI is not large enough to spoof the GPS receiver module of the target drone, it is not feasible for the attacker to conduct the GPS spoofing attack. The full list of equipment for each type of attack vector conducted by the adversary can be found in Table 4.1 (marked with ✓). It should also be noted that the adversary has no limitation on conducting different combinations of six attack types at the same time, using multiple pieces of available equipment.

Finally, it is important to note that software vulnerabilities are out of the scope of this study.

Attack Type / Equipment	**FAI** ([8],[37])	**GPS spoofing** ([21])	**Optical flow spoofing** ([21])	**EMI injection on SPI** ([19])	**EMI injection on Mag** ([21])	**Acoustic Noise Injection** ([21])
Signal generator	✓			✓		✓
Power amplifier	✓			✓		✓
High gain antenna	✓			✓		
Spectrum (logic) analyzer	✓			✓		
Oscilloscope	✓			✓		
HackRF One / GPS-SDR-SIM		✓				
ASNISH movie projector			✓			
5V Electromagnet - 25 kg Holding Force (P40/20)					✓	
Speaker						✓
DC Power	✓			✓		✓

Table 4.1: Equipment list for carrying out different PSAs and FAI attacks.

Chapter 5

Attack Methodology

The goal of this chapter is to first specify formally the attacker's requirements to capture the target drone without crashing it, using physical-layer attacks, namely PSAs and FAI attacks. We also introduce the motivation behind each attack requirement. Next, we discuss the rationale of the attack, and the attack framework algorithm is then proposed to address the requirements. At the end of this chapter, we present attack strategies to find how the identified attack requirements are fulfilled.

5.1 Attack Requirements

Table 5.1 introduces and motivates the attacker's requirements for successfully capturing a target drone model (as described in Section 4.1) without crashing it while exhausting the battery of the drone.

Attacker's Requirement	**Motivation**
R1. No crash occurs (before battery exhaustion)	Access valuable drone's resources (e.g., memory, sensitive information)
R2. Keeping the drone inside the ROI	The drone is more exposed to attack vectors, depleting energy more quickly
R3. A specification of the attacker ROI	No limitations on launching different attack types, solving the dependency issue
R4. Stealthiness No physical interaction No network connection	No possibility to interact with the drone through a standard communication channel due to the nature of attacks
R5. Prioritization of attack vectors	Adaptability to different drone's models, increasing the likelihood of a successful attack
R6. A fallback mechanism if a high-priority attack vector is not successful	Not constantly perform an unsuccessful high-priority attack vector

Table 5.1: A list of attacker's requirements and their corresponding motivations.

Specifically, we formulate six requirements that an attacker must provide to constitute a successful capture of a target drone. First, the adversary needs to ensure that the drone does not crash until its battery is exhausted, forcing the drone to land in the attack zone. Providing this requirement enables the attacker to safely access valuable target resources (e.g., sensitive data in the memory of the drone), which can become corrupted or inaccessible due to a sudden crash. The adversary should also force the victim drone to stay in a specific attacker's ROI, with the aim of launching attack vectors consistently, thus depleting energy more quickly. It is important to note that the attacker must also define a specification of their ROI for all attack vectors used. Further, the attacker must be stealthy and not establish any physical or network connection with the target drone, so as to address the existing research gap identified in this work (See Table 3.1). Moreover, the attacker must have this ability to dynamically prioritize attack vectors, so as to provide adaptability to different drone's models, increasing the likelihood of a successful attack. Finally, the adversary needs to have a fallback mechanism in case the high-priority attack vector fails.

5.2 Rationale of the Attack

As we mentioned in Section 3 (See Table 3.1), there exists a research gap in investigating how physical-layer attacks can be used to deplete energy of a target UAV (i.e., drone) as quickly or effectively as possible, satisfying attack conditions such as the stealthiness of the attack without establishing any physical interactions or network connections with the victim drone. To address this research gap, six different prioritized PSAs and FAI attacks can be conducted (using equipment listed in Table 4.1) based on their direct or indirect impacts on rotating the servo motors of the drone (See Table 3.3).

At the same time, as we discussed in Section 4, the ultimate aim of the attacker in this work is to capture the target drone without crashing it. To reach this main goal, the adversary should not only exhaust the battery of the drone effectively, eventually forcing it to land, but (s)he should also consider maintaining the drone in the attack zone (adversary's ROI) during the attack, depleting its energy quicker. In this section, we present a probabilistic attack framework algorithm for exhausting the energy of a drone (Algorithm 1), meeting our attack goals, as well as addressing our identified attack requirements (presented in Table 5.1).

In general, the proposed attack framework takes as input a list of six attack vectors, which are prioritized based on their direct or indirect impacts on rotating the servo motors of a target drone (See Table 3.3 for an example). Considering satisfying all the attack conditions and goals, this algorithm then returns a list of successful attack vectors with the highest probability, resulting in the exhaustion of the drone's battery, thereby forcing the drone to land (i.e., capture it). The details of Algorithm 1 are provided in the following section.

5.3 Attack Framework Algorithm

According to Line 1-6 in Algorithm 1, the preliminary step for the attacker is to identify the drone type and model using available drone detection techniques (e.g., [17], [35], [22]) before carrying out a list of m attack vectors, including FAI attack and PSAs. This preliminary step phase can help the attacker prioritize its attack vectors with respect to others based on the identified drone type and model (creating $Input\_modified[1, m]$), increasing the likelihood of a successful, effective energy exhaustion attack. On the other hand, if no drone model is detected, the default prioritized list of attack vectors ($Input[1, m]$) is used (e.g., prioritized attacks shown in Table 3.3).

In the next step (Line 8-25), the attacker selects and conducts the most power-consuming attack vector $[j]$ ($j = 1$ such that $1 \leq j \leq m$) from the prioritized list for at most two times, each time running a random T seconds. During each attack, the adversary monitors continuously whether the drone is crashing or not. To ensure that the attack is effective and successful in the first or second attempt, the adversary observes the impact of each attack on the drone. Noted

Algorithm 1 Proposed Attack Algorithm for Exhausting the Battery of a Drone

Attack Goals:

- Exhaust the battery of the drone effectively while preventing it from crashing
- Capture the drone by keeping it stay inside the attack zone border (adversary's ROI)

Attack Conditions:

- The stealthiness of the attack
- The attacker has no physical interaction
- The attacker has no network connection with the drone

Inputs:

- *Input*$[1, m]$: A list of m attack vectors which fulfill the above conditions and are prioritized based on their power consumption level from the highest to the lowest (i.e., Table 3.3)
- *Attack_zone border*: A specification of the attacker ROI

Output *Output_AttacksList*: A successful attack reaching its goals with the highest probability (An array of successful attack vectors conducted on the target drone)

if the drone model is found **then** //Pre-step attack phase for creating *Input_modified*
    $Input\_modified$[1,m] ← Modify the *Input*$[1, m]$ based on the model of the drone;
    Output_AttacksList ← Do_Attack($Input\_modified$[1,m], *Attack_zone border*);
else
    Output_AttacksList ← Do_Attack(*Input*$[1, m]$, *Attack_zone border*);
     //launch the default attack in case no model of the drone is found
end if
function Do_Attack(*Input*$[1, i]$, *Attack_zone border*)
    $j = 1$; // Define a counter to select from prioritized attack vectors
    $Final\_array$ = null; // The array of successful attack vectors conducted on the target drone
    for each attack vector [j] in the *Input*$[1, i]$ **do**
        Attack_attempt = 2; // Try conducting the attack vector [j] at most 2 times
        if the energy of the drone is exhausted without crashing it **and** the location of the drone remains inside the *Attack_zone border* **then**
            return $Final\_array$;
        else
            while *Attack_attempt* $\neq 0$ **and** drone does not crash **do**
                Select and conduct the highest priority attack vector[j] for a random T seconds;
                Update the location of the drone under the attack;
                if the desired impact of the attack vector is observed **then**
     // the observation can be different based on each attack type
                    Add the attack vector [j] to the $Final\_array$;
                else
                    Attack_attempt = *Attack_attempt* − 1;
                end if
            end while
            if *Attack_attempt* $= 0$ **and** drone does not crash **then**
                $j = j + 1$; // Update the counter to select the next prioritized attack vector
            end if
        end if
    end for
end function

that this observation can be different for each attack type (See Table 3.3 for examples). Also, the possible changes in the location of the victim drone during each attack are updated so as to ensure that the drone remains in the adversary's ROI. In case the selected attack vector $[j]$ is not effective after running two times and the drone still does not crash, the next highest priority attack vector is selected by updating the counter j (Line 25). The algorithm repeats the same actions described above for the new attack vector $[j]$ ($j = 2$ such that $2 \leq j \leq m$). Eventually, Algorithm 1 returns an array of successful attack vectors conducted on the target drone ($Final\_array$) only when the attack goals are met (i.e., satisfying both conditions in Line 12).

5.4 Attack Strategy

Table 5.2 shows the identified attacker's requirements in the previous section and how we can meet the requirements by proposing attack strategies, facilitating the exhaustion of a target drone's battery, and eventually capturing the drone.

Requirement	**Attack Strategy**
R1	Choosing a specific configuration (range) of *attack intensity*
R2	
R3	Defining a three-dimensional spatial area
R4	Applying PSAs and FAI attacks
R5	Using drone detection methods
R6	- Check if the desired impact of the current attack vector is observed - Define *Attack_attempt* variable

Table 5.2: Attacker's requirements and their corresponding attack strategy.

In order to provide requirements for executing PSAs and FAI attacks *effectively* against a target drone in a *controlled way* without any *crashes* (R1 & R2), we define a specific attack parameter, namely **attack intensity**, which controls the intensity of the impacted (spoofed) variables for each attack vector in our attack framework (See Table 3.3). For example, the attack intensity for GPS spoofing implies how much positioning error this attack can cause. By choosing a specific configuration (range) of attack intensity value for each attack vector, the target drone does not crash and stays inside the adversary's ROI while the drone's battery is more quickly depleted, causing the exhaustion of the battery in an effective way, eventually forcing the target drone to land (i.e., capture it). Note that the details of the attack intensity parameter and its corresponding range of values for each attack vector are explained in Chapter 6.

Regarding the attacker's ROI (R3), the attacker should be able to specify a three-dimensional spatial area (i.e., a spherical shape) with the same specific value for longitude, latitude, and altitude. All attacks can be conducted within this area using the equipment shown in Table 4.1.

The *stealthiness* of the attack (R4) is satisfied since the adversary utilizes energy depletion attacks, which do not require any physical interaction, as well as the establishment of any network connection with the target drone. In fact, the adversary ensures to be stealthy by applying specific types of attack vectors, known as physical-layer attacks, such as PSAs and the FAI attacks (See Table 3.3). These attack types can exploit limitations and vulnerabilities of sensors for distinguishing between correct signals and spoofed signals [41].

Moreover, the attacker should be able to prioritize attack vectors with respect to others based on their energy consumption level from the highest to the lowest. To meet this attack requirement (R5), the adversary came up with a strategy to try to understand which attack(s) he can deploy based on the drone type and model. The attacker's strategy thus is to either identify the drone type and model in a preliminary step phase using available drone detection methods (e.g., [17], [35], [22]) or choose a default prioritized attack list for a well-known drone model (e.g., DJI model) in the real-world environment. The attacker can thus exploit the detected drone specifications and data sheets to better understand the specific available sensors, the version of flight controller firmware, and the type of fail-safe algorithm used in the flight controller. This useful information can then be used by the attacker to adjust attack vectors, increasing the likelihood of a successful attack vector, thereby depleting energy quicker. For example, the new DJI drone models, which have the latest firmware version, can be more robust against GPS spoofing attacks [11]. Thus, the attacker can give lower priority to the GPS spoofing attack in the proposed attack framework algorithm to exhaust the drone's battery.

Finally, the attacker should be able to select and conduct a less powerful attack if the current attack vector is not effective on the target drone (i.e., a fallback mechanism). To meet this requirement (R6), the attacker's strategy is to verify whether the desired impact of the current attack vector is observed or not on the target drone.

Chapter 6

Experimental Results

In this chapter, we discuss both the experimental setup and experiment settings, followed by the general plan for conducting the experiments. Then, we present the results of the experiments to evaluate the impacts of PSAs and FAI attacks on the energy consumption of a target drone.

6.1 Experimental Setup

In our experiments, we select a popular open-source flight controller software, namely *ArduPilot* [43], and in particular the latest version available at the time of this writing (V4.7.0-dev). The Quadcopter model in ArduPilot considered for this study is known as ArduCopter [44], and we use ArduCopter SITL simulator [45] for conducting the experiments. In general, the SITL simulator can simulate a wide variety of vehicle types (e.g., plane, copter, rover) along with their components (e.g., IMU sensors, GPS receiver, vision sensors, battery). In fact, SITL enables us to run the same ArduPilot firmware code that would be used on a real drone directly on a PC or a laptop without any hardware. To run ArduCopter SITL, a tool named *sim_vehicle.py* is executed. With this tool, MAVProxy, which is an open-source GCS based on the MAVLink protocol, is also automatically started. The ArduCopter simulation platform can also simulate some specific environmental conditions (e.g., wind speed or direction) that affect the power consumption of the UAV as well [2]. We focus on a specific type of ArduCopter, which has four servo motors (i.e., quadcopter [46], [18]). We run the simulations on an Ubuntu 22.04.5 LTS (Jammy Jellyfish) 64-bit HP Zbook Studio G5 laptop with Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz processor and 32 GB RAM.

It should be noted that there are other popular open-source RV software such as PX4 [49] and Paparazzi [54]. However, we chose to use ArduPilot because it is the largest open-source autopilot software suite deployed in over one million UAVs [15], [6], and supports a wide variety of RV hardware platforms (e.g., Pixhawk, Bebop, Navio, etc.) [42].

Also, we select the default configuration parameters of ArduCopter in our experiments [47]. Specifically, the simulated quadcopter has three separate IMUs, each containing one accelerometer and one gyroscope sensor. In addition, the quadcopter flight controller in the SITL simulator uses by default the latest version of EKF sensor fusion algorithm (i.e., EKF3). It is important to note that the number of EKFs running in parallel is equal to the number of IMUs. So, there are three EKF cores inside of the flight controller, each using a different IMU.

Further, we take into account a class of widely used actuators in drones, specifically servo motors that use PWM to encode actuation data (e.g., rotation angle or speed) [8, 48]. In our experiments, the simulated quadcopter possesses four PWM-controlled servo motors.

Finally, all the simulation logs from ArduCopter SITL simulator are sampled every 100 ms (10 Hz) from the beginning of each experiment.

6.2 Experiment Settings

Before running the ArduCopter SITL simulator (i.e., executing sim_vehicle.py), we add a specific simulation startup option, namely *speedup*, to boost our experimental speedup. Note that this option directly controls the value of SIM_SPEEDUP parameter in SITL. Although the default value for the speed of each simulation is 1, we set *speedup* to 4 in this work to decrease the time of our simulation, getting the experimental results four times faster. After running SITL, we run MAVLink commands in the MAVProxy Command Prompt, so as to switch the flight status of the simulated quadcopter to "Guided" mode, and then it arms and starts to take off, hovering at an altitude of 3 meters. This process takes 20 seconds in the SITL simulator before our attack experiments are executed. Table 6.1 summarizes the SITL configuration that we set before executing our attack experiments.

Related SITL Parameter / MAVLink Commands	**Description**	**Our Configuration**
SIM_SPEEDUP	Runs the simulation at multiples of normal speed, increasing the simulation speedup	Add *speedup* = 4 to *sim_vehicle.py*
mode n	Changing the flight mode to n	mode guided
arm throttle	Arm the throttle of the motor	arm throttle
takeoff m	Take off to an altitude of m meters	takeoff 3

Table 6.1: Our general experiment settings before running attack experiments in ArduCopter SITL.

It is important to note that the simulated quadcopter by default does not have an optical flow sensor. So, it is necessary to add this sensor into ArduCopter SITL before evaluating specifically the impact of optical flow spoofing attack on the total energy consumption of the quadcopter. note that we leverage the same experiment settings used by [21] for adding the optical flow sensor into our simulation platform. According to the implementation of the optical flow sensor in [21], we need to configure simulation parameters for adding a rangefinder sensor in addition to the parameters related to the optical flow sensor. Table 6.2 summarizes a list of related simulation parameters that we use and configure to add a rangefinder and optical flow sensor in SITL. After executing the mentioned simulation parameters in SITL using our code, we run *reboot* MAVLink command in MAVProxy, so as to restart the ArduPilot SITL simulator with enabled optical flow sensor enabled.

For the sake of simplicity, we consider in our experiments that the simulated quadcopter is located in the center of the attacker's ROI, which is a 3D-spatial area with a fixed value of 20000 degE7 ($degrees \times 10^7$) for both latitude and longitude, while the altitude is 20 m.

To implement PSAs and FAI attacks in our simulation platform, we need first to identify and then modify the SITL configuration parameters that are affected directly by conducting the attacks in the real-world experiments. Table 6.3 summarizes the identified simulation parameters with our configuration of such parameters to model the attack experiments in SITL. It is important to note that our attack experiments need to meet all the identified attacker's requirements introduced in Section 5.1 (See Table 5.1). To ensure that the drone does not crash before battery exhaustion and it remains inside the ROI, the attack strategy is to define and configure a specific range of *Attack Intensity* (See Table 5.2). In fact, the attack intensity implies the intensity of affected variables by conducting the attack on the target quadcopter. Note that Table 3.3 shows these impacted variables in more detail for each attack type. It should also be noted that the attack intensity is a value higher than 0. When this value equals 0, no attack experiment is performed; the drone thus continues its normal flight condition (i.e., hovering at its location at the altitude of 3 m).

To obtain some statistical variability in our experimental results, we consider two additional

SITL configuration parameters (presented also in Table 6.3): 1) weather conditions, which allow us to simulate wind speed, ranging from 0 to 15 m/s in our experiments. So, the simulation results of the simulated quadcopter energy consumption are closer to the results obtained in a real-world setting. It should be noted that we omit to consider wind speed values higher than 15 m/s because they simulate unreasonable extreme weather conditions (e.g., near gale, storm, and violent storm) [4], causing the drones to find it difficult to take off and complete its mission in the real world environment [2]. Another weather condition that we assume in our simulation experiments is the wind direction. In fact, we consider allowing random wind direction by changing between 1 and 360 degrees every 0.2 seconds. 2) The motor thrust value has a value between 0 and 1, and this value can affect the energy consumption of the simulated quadcopter. So, we consider changing the thrust motor randomly every 1 s in our experiments to achieve random movements, resulting in different energy consumption.

Related SITL Parameter	**Description**	**Our Configuration**
SIM_SONAR_SCALE	Sonar conversion scale from distance to voltage Unit: meters per volt	Set it to 10
RNGFND1_TYPE	Type of connected rangefinder	Set it to 1 (i.e., Analog)
RNGFND1_SCALING	Scaling factor between rangefinder reading and distance Unit: meters per volt	Set it to 10
RNGFND1_PIN	Analog or PWM input pin that rangefinder is connected to	Set it to 0
RNGFND1_MAX_CM	Maximum distance in meters that rangefinder can reliably read Unit: meters	Set it to 5000
RNGFND1_MIN_CM	Minimum distance in meters that rangefinder can reliably read Unit: meters	Set it to 0
SIM_FLOW_ENABLE	Enable Optical Flow sensor	Set it to 1
FLOW_TYPE	Optical flow sensor type	Set it to 10 (i.e., SITL)
EK3_SRC1_VELXY	Velocity Horizontal Source	Set it to 5 (i.e., Optical Flow)
EK3_SRC1_POSXY	Position Horizontal Source	Set it to 0 (i.e., None)
EK3_SRC_OPTIONS	EKF Source Options	Set it to 0 (i.e., Fuse All Velocities)
EK3_SRC1_POSZ	Position Vertical Source	Set it to 0 (i.e., Fuse All Velocities)
EK3_SRC1_VELZ	Velocity Vertical Source	Set it to 0 (i.e., None)
EK3_SRC1_YAW	Yaw Source	Set it to 1 (i.e., Compass)

Table 6.2: The related SITL parameters implemented by [21], and we also use them in this work to add an optical flow sensor in ArduCopter SITL before conducting optical flow spoofing.

SITL Parameters	Description	Our Configuration
SERVO1/2/3/4_FUNCTION	Function assigned to each servo motor no.1/2/3/4	Set it to 0 for controlling the servo output by MAVLink command: MAV_CMD_DO_SET_SERVO. This command is used to control FAI attack intensity by set a servo to a spoofed PWM value, between [1000-2000] microseconds (μs)
SIM_ACC1/2/3_RND	Simulated motor vibration caused by acceleration noise for 3 Accelerometers in (m/s^2)	Set these parameters to higher than 0 for controlling the intensity of physical sensor attacks. Vary in a limited range, depending on the attack type tested for our attack experiments (See Table 6.5)
SIM_ACC1/2/3_BIAS_X(Y)(Z)	Bias of 3 Accelerometer sensors for X/Y/Z-axis (i.e., Xacc, Yacc, Zacc)	
SIM_GYR1/2/3_RND	Simulated motor vibration caused by gyroscope noise for 3 gyroscopes	
SIM_GYR1/2/3_BIAS_X(Y)(Z)	Bias of simulated Gyroscope sensor for X/Y/Z-axis (i.e., Xgyro, Ygyro, Zgyro) in rad/s	
SIM_MAG_RND	Simulated motor vibration caused by magnetometer noise	
SIM_FLOW_RND	Optical flow sensor measurement noise (rad/s)	
SIM_FLOW_DELAY	Optical flow data delay (ms)	
SIM_GPS1_NOISE	Amplitude of the GPS altitude error (m)	Set these parameters to higher than 0 for controlling the intensity of GPS spoofing attack. These parameters are controlled by *Attack Intensity*
SIM_GPS1_VERR_X(Y)(Z)	GPS Velocity Error Offsets in NED (X/Y/Z-axis) (m/s)	
SIM_GPS1_GLTCH_X(Y)(Z)	Glitch offsets of simulated GPS receiver (X/Y/Z-axis) (m)	
SIM_GPS1_NUMSATS	Number of GPS satellites	Set it to 5 (the default value = 10)
SIM_WIND_SPD SIM_WIND_DIR	Simulate weather conditions (i.e., Wind speed & direction)	wind speed: from 0 to 15 m/s wind direction: randomly changing between [1-360] degrees every 0.2 s
SIM_ENGINE_MUL	Motor thrust value of the target quadcopter	- Add randomness in Motor Thrust: changing between [0-1] every 1 s

Table 6.3: The configuration of SITL parameters identified and modified in this work to simulate our attack experiments, adding both weather conditions and motor thrust randomness to provide the statistical variability.

6.3 Experiment Plan

The attack experiments that we conduct in this work are divided into two main categories: i) the 1-minute attack experiment, in which the attack duration is 1 minute, and ii) the full-battery attack experiments, in which there is no limit on the attack duration and the experiments keep running until the simulated drone is forced to land due to the battery exhaustion (i.e., battery level of the simulated quadcopter reaches 5% of its total capacity, in line with many research papers [39], [24], and user manuals, e.g., the 3DR SOLO [1]).

Table 6.4 describes the specific attack types tested in this work for each category of 1-min and full-battery attack experiments.

Experiment Category / **Attack Type**	**1-min attack**	**Full-battery attack**
FAI	✓	✓
GPS Spoofing	✓	✓
Optical flow spoofing	✓	✓
Acoustic Noise Injection	✓	✓
EMI Injection on SPI	✓	✓
EMI Injection on Mag	✓	×
Accelerometer Spoofing (Xacc/ Yacc/ Zacc spoofing)	✓	×
Gyroscope Spoofing (Xgyro/ Ygyro/ Zgyro spoofing)	✓	×
Coupling Attack (Zacc + Zgyro spoofing)	✓	×
Coupling Attack (Acoustic Noise Injection + GPS spoofing)	×	✓
Coupling Attack (FAI + GPS spoofing)	×	✓

Table 6.4: A comparison of different attack types simulated in this work for both 1-min and full-battery experiments.

In fact, we plan to simulate the impact of PSAs and FAI attack types executed separately in 1-min attack experiments without running any combinations of two different attack types together. This plan can not only provide us with a better analysis of the impact of each attack on the energy consumption of the simulated quadcopter, but it can also enable us to select and combine attack types with the aim of exhausting the battery more effectively. As we will discuss in more detail in Sec.6.5, we exclude testing the EMI attack on Mag sensor in full-battery experiments since this attack type does not have any negative impacts on the total energy consumption of the simulated quadcopter in our 1-minute experiments. We also simulate the impact of acoustic noise injection on IMU sensors (accelerometers and gyroscopes), both separately and together in 1-min experiments. Based on our observed results, we notice that the combination of spoofing both accelerometers and gyroscopes supersedes each separate IMU sensor spoofing attack. So, we choose only to evaluate the impact of spoofing both accelerometers and gyroscopes at the same time in full-battery experiments.

Before running each attack type presented in Table 6.4, we manually switch the flight status of the quadcopter to "Guided" mode, and it is armed and taken off to maintain its position (hovering) at 3 meters. After 20 seconds, we run each attack type for 1 minute (i.e., 1-min attack experiments) or until battery exhaustion (i.e., full-battery attack experiment). It is also important to note that, according to Table 3.2 and Section 4.2, we conduct both 1-min and full-

battery attack experiments for each type of PSAs based on their capabilities that have "claimed impacts" on the target quadcopter. However, we perform a specific type of GNSS spoofing attack (i.e., GPS spoofing) and exclude simulating GNSS spoofing attack because we chose the default configuration parameters of ArduCopter SITL that utilizes only GPS signals for GNSS system configuration (i.e., the configuration parameter GPS1_GNSS_MODE = 0 by default) [47].

To assess the impact of acoustic noise injection on energy consumption of the simulated quadcopter in 1-min attack experiments, we consider spoofing IMU sensors (accelerometers and gyroscopes) in three following cases: i) only when accelerometer values are spoofed (Xacc, Yacc, Zacc), ii) only when gyroscope values (Xgyro, Ygyro,Zgyro) are spoofed, and iii) when a combination of accelerometer and gyroscope sensor values (i.e., Zacc and Zgyro) are manipulated at the same time, which is called a coupling attack (See Table 6.4).

Focusing on attack types for the full-battery attack experiment category, we conduct all the PSAs and FAI attacks in our simulation platform, except EMI injection on the Mag sensor. The details of this exception are shown in Section 6.5.1. We also implement two coupling attacks in this experiment category, in which the GPS spoofing attack is combined with either acoustic noise injection or FAI attack.

6.3.1 Attacks Implementation

Table 6.5 classifies the SITL parameters and MAVLink commands used for simulating PSAs and FAI attacks in both 1-min and full battery attack experiments, as implemented by [21] and this work.

Specifically, we utilize the same methodology used by [21] for simulating acoustic noise injection, GPS spoofing, EMI attack on mag sensor, and optical flow spoofing in ArduPilot SITL, using Pymavlink library v2.4.47 [55] that enables us to not only communicate with the simulated quadcopter via MAVLink protocol [25], but also modify SITL simulation parameters by sending specific MAVLink commands (MAV_CMD_DO_SET_SERVO).

Regarding the simulation of FAI attacks, we could not find any existing simulation code in the literature [8], [37] to demonstrate the impact of FAI attacks in ArduPiloT SITL. So, we wrote our own Python script codes, using the Pymavlink library mentioned earlier, so as to simulate the impact of FAI attack on one of the servo motors of the target quadcopter. For example, we first changed SERVO1_FUNCTION to 0, to control the servo motor 1 output by MAVLink Command MAV_CMD_DO_SET_SERVO. We then use this command to manipulate the PWM signal duration, enabling us to model the impact of FAI attack in SITL.

It should be noted that we also analyze separately the impact of spoofing accelerometers and gyroscopes on the energy consumption of the simulated quadcopter so that we can evaluate how each sensor spoofing can influence the quadcopter's energy consumption. For simulating accelerometer and gyroscope spoofing attacks in our simulation platform, we consider modifying SIM_ACC1/2/3_BIAS_X(Y)(Z) and SIM_GYR1/2/3_BIAS_X(Y)(Z), respectively. Note that we also set the parameters SIM_ACC1/2/3_RND and SIM_GYR1/2/3_RND to 400 in both accelerometer and gyroscope spoofing attacks. The result of this analysis is given as a part of 1-min attack experimental results in the next section.

While the authors in [21] use Gazebo simulator to simulate EMI attacks on SPI, we perform this attack in ArduPiloT SITl, using the identified STIL simulation parameters that affect all the sensor values at the same time (see Table 6.5)

As we discussed in Section 6.2, we use two additional SITL parameters (See Table 6.3) for providing statistical variability in our attack experiments. So, we added new lines of codes into both 1-min and full-battery attack experiments to 1) simulate different weather conditions, and 2) simulate the motor thrust randomness. We also defined a specific function in our code to check if the drone is inside the attacker's ROI or not.

Reference for Attack Implementation	**Attack Type**	**Related SITL Parameter/ MAVLink Command**
[21]	Acoustic Noise Injection	SIM_GYR1/2/3_RND SIM_ACC1/2/3_RND
	GPS spoofing	SIM_GPS1_NOISE SIM_GPS1_VERR_X(Y)(Z) SIM_GPS1_GLTCH_X(Y)(Z) SIM_GPS1_NUMSATS
	EMI on Mag	SIM_MAG_RND SIM_ACC1/2/3_RND
	Optical flow spoofing	SIM_FLOW_RND SIM_FLOW_DELAY
This work	FAI	SERVO1/2/3/4_FUNCTION (= 0) MAV_CMD_DO_SET_SERVO
	Accelerometer spoofing	SIM_GYR1/2/3_RND (= 400) SIM_ACC1/2/3_RND (= 400) SIM_ACC1/2/3_BIAS_X(Y)(Z)
	Gyroscope spoofing	SIM_GYR1/2/3_RND (= 400) SIM_ACC1/2/3_RND (= 400) SIM_GYR1/2/3_BIAS_X(Y)(Z)
	EMI on SPI	SIM_GYR1/2/3_RND SIM_ACC1/2/3_RND SIM_MAG_RND

Table 6.5: The classification of SITL parameters and MAVLink commands used for simulating PSAs and FAI attacks in ArduPilot SITL.

6.4 Performance Metrics

Table 6.6 introduces the main performance metrics that we use based on each category of 1-min and full-battery attack experiments.

Experiment Category / **Metrics**	**1-min attack**	**Full-battery attack**
Total Average Energy Consumption (hJ)	✓	×
Average Battery Depletion Percentage (%)	✓	×
Average Exhaustion Time (s)	×	✓

Table 6.6: Performance metrics used in our experiments.

According to Table 6.6, we use *Total Average Energy Consumption* and *Average Battery Depletion Percentage* as two performance metrics to evaluate 1-min attack experiments. The term *average* is used in these metrics since we conduct 1-min experiments for six different wind speed values (i.e., 0, 5, 7, 10, 13, 15 m/s), calculating the total energy consumption in each experiment for each value of the wind speed separately. It should be noted that ArduCopter SITL measures and reports the total energy consumed of a simulated quadcopter's battery in hJ (hecto-Joules), which is equal to 100 J. To calculate how much percentage of the battery is depleted corresponding to its total energy consumption, we leverage the following formula [32]:

$$\text{Battery Depletion Percentage}(\%) = \frac{\text{Total Energy Consumption } (J)}{\text{Battery Capacity } (J)} \times 100$$

Note that the default SITL configuration parameters for the battery capacity of a simulated quadcopter (i.e., BATT_CAPACITY) and its battery voltage (SIM_BATT_VOLTAGE) are 3300 mAh and 12.60 V, respectively. So, the following equation is used to convert the battery capacity to Joules [40]:

$$\begin{aligned}\text{Battery Capacity}(J) &= \frac{\text{Battery Capacity}(mAh)}{1000} \times \text{Battery Voltage}(V) \times 3600 \\ &= 3300 \times 12.60 \times 3.6 = 149688\end{aligned}$$

The metric that is used for assessing full-battery experiments is *Average Exhaustion Time* in seconds (s). As previously mentioned, many drones in real-world environments land when their battery capacity reaches 5% ([39], [24], [1]). Taking this into account for ArduCopter SITL (in which the simulated quadcopter has 1496.88 hJ battery capacity by default), the corresponding total energy consumption of the quadcopter, which spends 95% of its battery capacity, is 1422 hJ. We use this specific value in our code to determine the exhaustion time of the battery in the full-battery experiments. Note that we also test full-battery experiments for three different wind speed values (i.e., 0, 7, 15 m/s), meaning that we calculate the *Average Exhaustion Time* for each full-battery attack experiment.

6.5 Simulation Results

This section shows the results obtained from both 1-min attack and full-battery experiments for each one of the attack types presented in Table 6.4. We present the results about each attack type tested for 1-min attack experiments (Section 6.5.1), and full-battery experiments (Section 6.5.2). As mentioned in the experimental setup of this work (Section 6.1), we consider using the default configuration parameters of ArduCopter with the latest version available at the time of this writing (V4.7.0-dev) in both 1-min and full-battery experiments. This consideration enables us to have a fair comparison in our simulation results.

6.5.1 One-minute Attack Experimental Results

In the following subsections, we investigate the impact of various attack types (described in Table 6.4) on two related performance metrics used in our 1-min attack experiments (see Table 6.6).

It should also be noted that we evaluate these two performance metrics under no attacks for 1 minute when the simulated quadcopter is hovering at 3 m of altitude after arming and taking off the land (takes 20 s) for six different wind speed values. The results for total average energy consumption and average battery depletion percentage are 282.67 hJ and 18.88%, respectively. We present these two measurements in the following experiments when the attack intensity is 0.

Exp.1: FAI attack.

Figure 6.1 shows the results of the FAI attack, considering spoofed PWM signal duration in microseconds (μs) as the attack intensity. Note that in a normal flight condition in which the

drone is hovering at 3 m of altitude and the FAI attack is not running (attack intensity = 0), the observed PWM signal duration for a simulated quadcopter is 1590 μs.

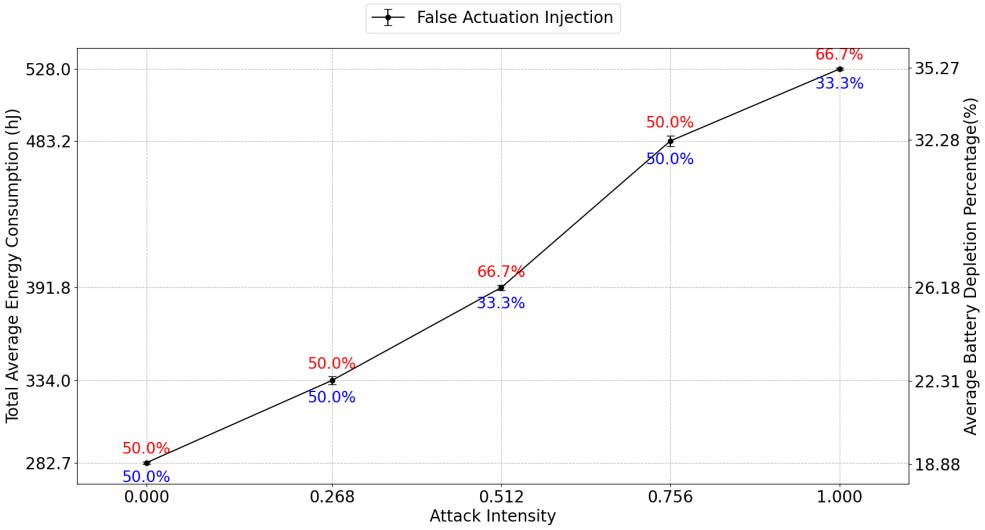


Figure 6.1: Impact of the FAI attack on both the total average energy consumption of a simulated quadcopter and its corresponding average battery depletion for a 1-min attack experiment.

As shown in Fig. 6.1, the highest total average energy consumption (528 hJ), and the corresponding average battery depletion percentage (35.27%) are obtained when the attack intensity is equal to 1, meaning that the spoofed PWM signal has its own maximum value (i.e., 2000 μs). Comparing the average battery depletion percentage observed for no-attack condition (i.e., 18.88%) with the one calculated by conducting 1-minute FAI attack with the maximum attack intensity, we notice an over 16% increase in battery depletion.

It is also important to note that the red number for each attack intensity in Fig.6.1 (e.g., 66.7% when the attack intensity is 0.512) represents the percentage of attack experiments that obtained total energy consumption equal to (or higher) than the corresponding total average energy consumption. Similarly, the blue number for each attack intensity indicates the percentage of attack experiments that result in total energy consumption lower than the average total energy consumption. Note that we consider obtaining these percentages in other attack experiments as well, which have their own attack intensity rather than spoofed PWM signals in FAI attack.

Exp.2: GPS Spoofing.

Figure 6.2 reports the result of our 1-min attack experiment, with specific reference to the impact of the GPS spoofing attack. To model the effect of the GPS spoofing attack in SITL using our code, we define the variable *Attack Intensity*, which controls all related SITL parameters (see Table 6.5).

We observe that the simulated quadcopter starts vibrating quickly inside the attacker's ROI after increasing the attack intensity from 0 to higher values (e.g., 1). This vibration specifically causes the drone to activate the EKF switching, besides moving up and down continuously (from 3 m to 2 m) inside the attacker's ROI. Note that EKF switching is a sensor fusion design that switches the primary EKF to the secondary EKF if the primary EKF shows the poorest error score among EKFs, allowing excluding a sensor that returns erroneous measurements. If we select the attack intensity value such that it is higher than 1.167, the drone crashes immediately in less than 1 minute period due to the activation of default EKF fail-safe behavior in ArduCopter SITL (i.e., switching flight mode to the manually-controlled LAND mode, causing the drone lands and eventually disarms its motors). As seen in Fig. 6.2, the highest observed total average

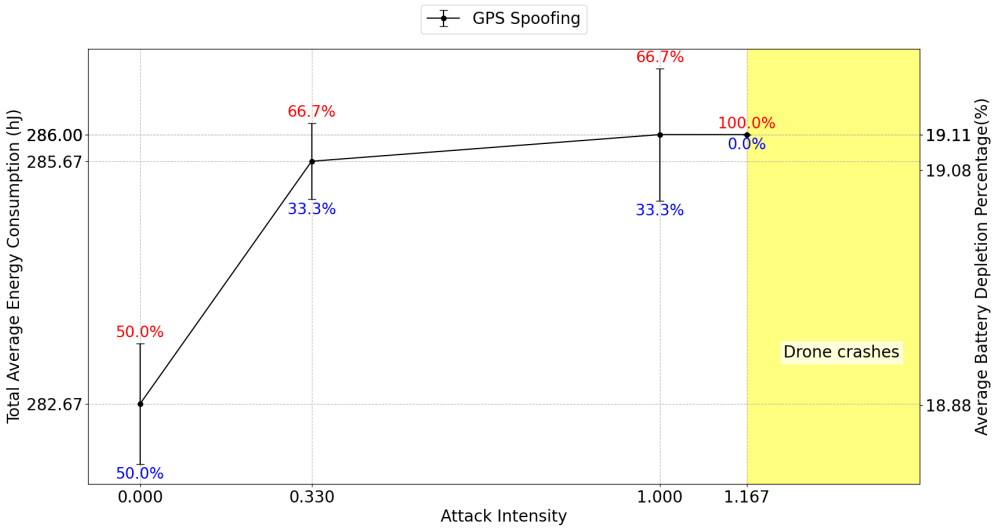


Figure 6.2: Impact of GPS spoofing attack on both total average energy consumption of a simulated quadcopter and its corresponding average battery depletion for a 1-min attack experiment.

energy consumption is 286 hJ, which is almost half of the same performance metric for FAI attack experiment (528 hJ). Focusing on the average battery depletion percentage, we noticed a very low increase (i.e., about less than 1%) in this metric compared to the situation without any attacks.

Exp.3: Optical Flow Spoofing.

We report in Fig.6.3 the result of experiments investigating the impact of optical flow spoofing attack. Similarly to other attack experiment results, we noticed that the higher the intensity of attack, the more total energy consumed, thereby causing more battery depletion. As seen in this figure, the threshold value of the attack intensity that led the drone to leave the attacker's ROI without any crashes in less than 1 minute is 0.55.

Exp.4: EMI on SPI.

Figure 6.4 shows how EMI attack on SPI channels can affect the total energy consumption and its corresponding battery depletion percentage in our 1-min experiments. The highest total energy consumption (285.83 hJ) is observed when we choose 1.125 for the attack intensity. Comparing the results of this attack with the ones for FAI attack shows how FAI attack, which directly affects each PWM-controlled servo motor of the simulated quadcopter, is more powerful than EMI attack on SPI. It should also be noted that when the attack intensity is higher than 1.125, ArduPilot activates a series of fail-safe logic, including yaw reset, EKF switching, GPS glitch fail-safe, and vibration fail-safe, altogether triggering the default EKF fail-safe action, eventually causing the drone to crash.

Exp.5: EMI on Mag.

After modeling this attack on our simulation platform using the related SITL parameter in Table 6.5, we observe that this attack fails to cause any negative impacts on the drone's attitude, thereby more energy consumption. This observation is also aligned with the results found in the recent paper [21] in which they showed that both *ArduPilot software versions* and *configuration parameters* related to the sensor filtering (e.g., range filter) and sensor fusion algorithms (e.g., EKF3) play a key role in filtering out the spoofed sensor values, causing no negative symptoms

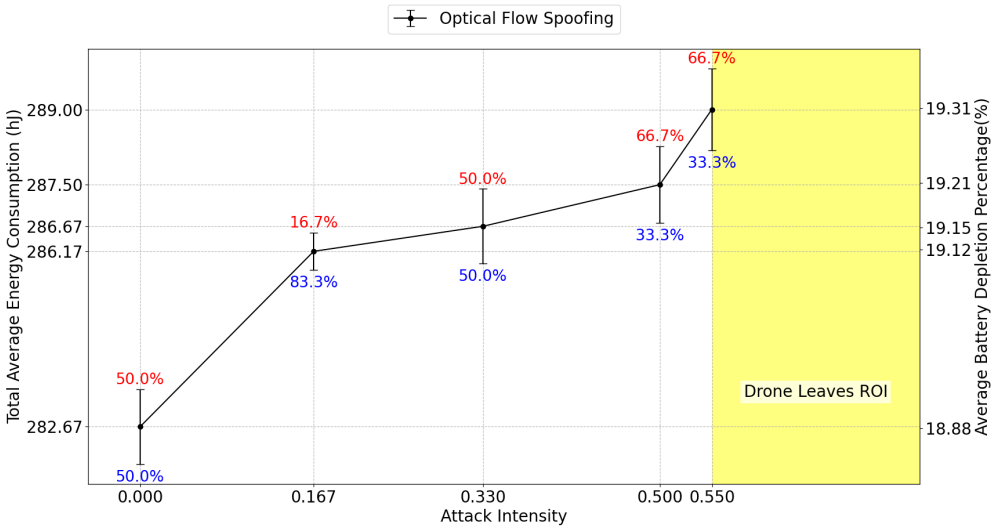


Figure 6.3: Impact of optical flow spoofing attack on both total average energy consumption of a simulated quadcopter and its corresponding average battery depletion for 1-min attack experiment.

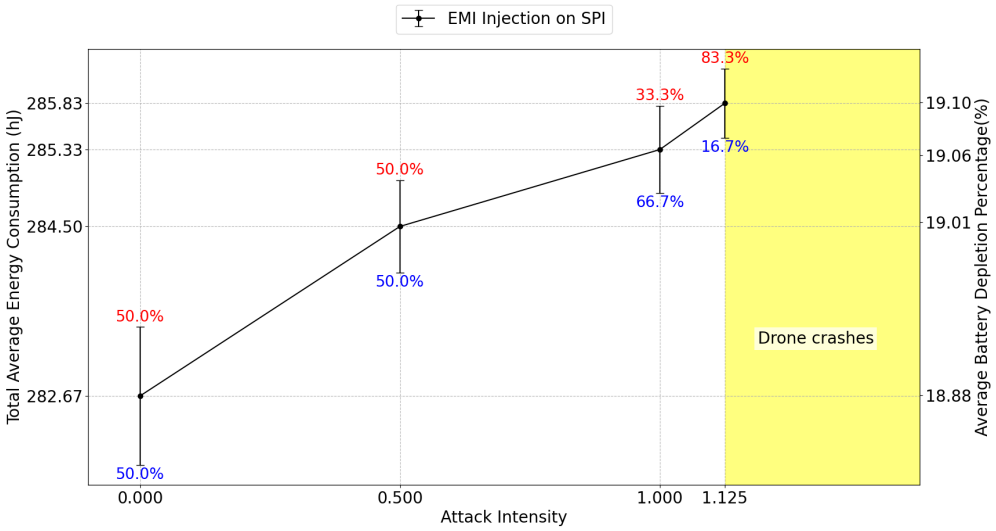


Figure 6.4: Impact of EMI attack on SPI channels on both total average energy consumption of a simulated quadcopter and its corresponding average battery depletion for 1-min attack experiment.

under the EMI attack on magnetometer sensors. Note that according to [21], the configuration parameter, namely EK3_MAG_M_NSE determines the magnetometer sensor's weight for the EKF3 sensor fusion algorithm used in ArduPilot. To enable and observe the impact of this attack experiment, we need to adjust this configuration parameter. However, we consider the default configuration parameters in all our experiments, so as to have a fair comparison.

Since this type of attack is not effective in our 1-min attack experiments, we specifically exclude this attack from the list of attack types tested for full-battery experiments.

Exp.6: Acoustic Noise Injection - Accelerometer Spoofing.

As we discussed earlier in Sec.6.3, we evaluate the impact of acoustic noise injection for 1-min attack experiments in three separate cases, namely spoofing i) solely available accelerometers, ii) solely available gyroscopes, and iii) a combination of both accelerometers and gyroscopes at the same time, which we consider it as a coupling attack.

Figure 6.5 presents how spoofing each one of three available accelerometer sensor values, namely Xacc, Yacc, and Zacc can influence both the total energy consumption and its corresponding average battery depletion percentage in our simulation platform.

The results of three experiments in Fig.6.5 also show that although increasing the attack intensity (i.e., spoofed bias values) causes more total energy consumption, this change has a limited impact on battery depletion percentage. Similar to other attack experiments except FAI attack, less than 1% of battery is depleted in this attack before the drone crashes or leaves the attack zone in less than 1 minute.

Exp.7: Acoustic Noise Injection - Gyroscope Spoofing.

We also investigate the impact of gyroscope spoofing with respect to three sensor values (Xgyro, Ygyro, and Zgyro) during a 1-minute attack duration. We report the results of our investigation in Fig 6.6.

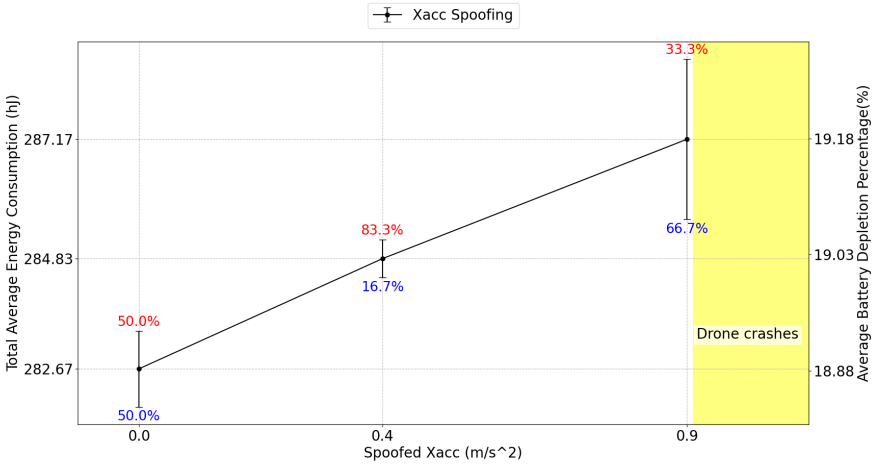
Exp.8: Acoustic Noise Injection - Zacc + Zgyro spoofing.

In this experiment, we consider a combination of spoofing simultaneously two distinguished sensor values on the same axis, namely Zacc and Zgyro (See Fig.6.5. (c) and Fig.6.6. (c)). Specifically, we compare how fixing the spoofed Zacc sensor value and changing the spoofed Zgyro in each experiment can affect both the total average energy consumption and its corresponding average battery depletion percentage. The results of this experiment are shown in Fig. 6.7

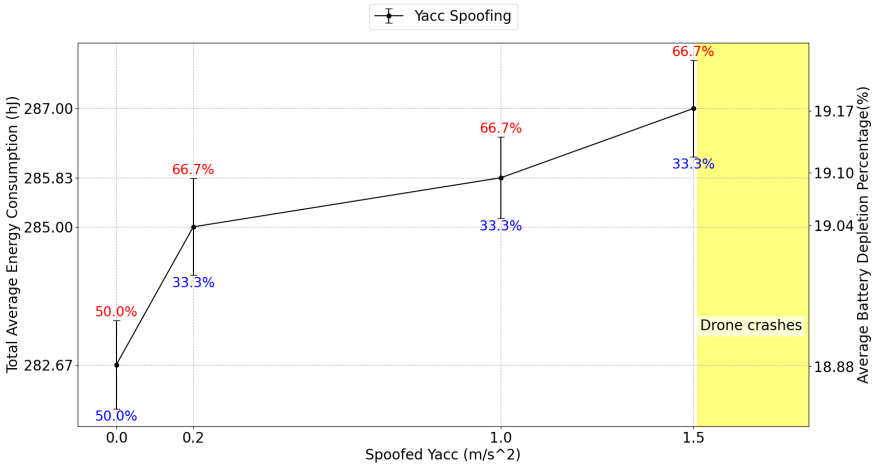
We notice a small increase in both performance metrics for both cases of coupled attacks, denoted by olive and grey lines in the figure. When we choose the spoofed Zacc = 0.7 m/s^2 and the spoofed Zgyro = 1 millrad/s, the simulated quadcopter crashes (depicted by a red star point in Fig 6.7). The drone hit also the ground for spoofed Zacc values higher than 25 millrad/s (yellow area in the figure).

Key Takeaways.

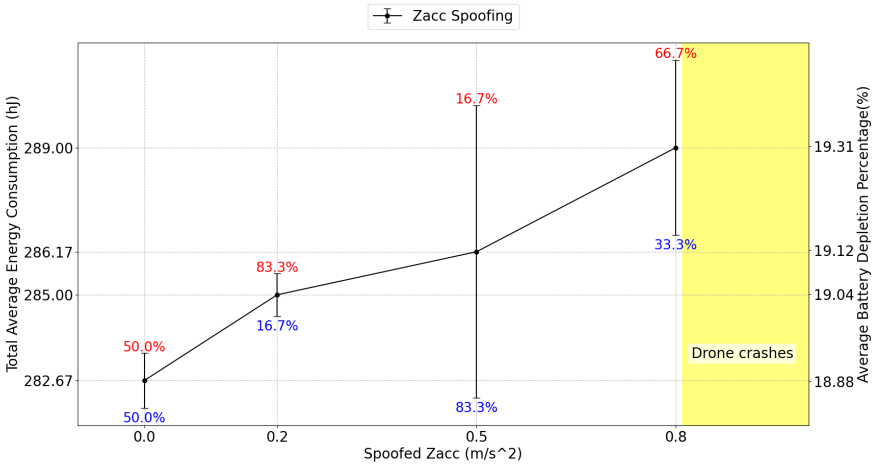
In summary, we conclude that the most successful attack type in our 1-min attack experiments is the FAI attack due to its direct impact on the speed and rotation of a servo motor, causing the quadcopter to consume more energy, thereby depleting its battery more quickly. In contrast, other attack types tested in 1-min experiments have a comparable limited impact on increasing the total energy consumption of the drone, and its corresponding battery depletion percentage. We also found that optical flow spoofing is the only attack in our 1-min experiments that can cause the simulated quadcopter to leave the attacker's ROI without crashing for attack intensities higher than 0.55. So, it is challenging for the attacker to keep the simulated drone inside the attack zone, by using only optical flow spoofing.



(a) Impact of spoofing the Xacc sensor values of all three available accelerometers.

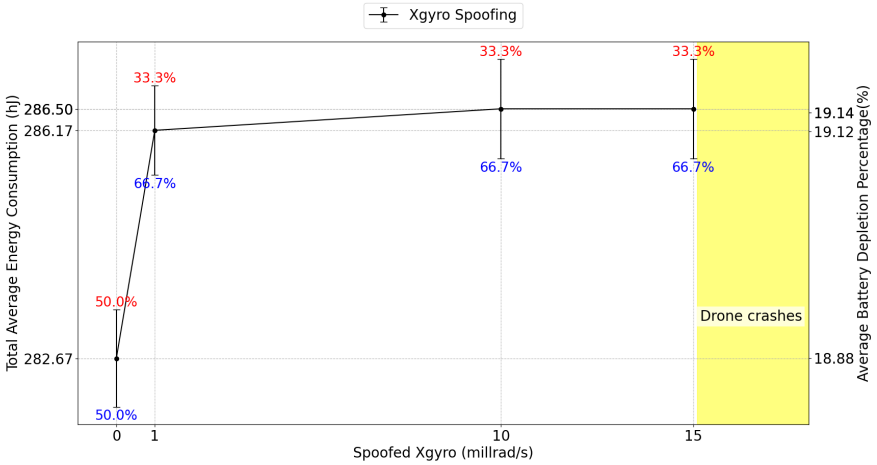


(b) Impact of spoofing the Yacc sensor values of all three available accelerometers.

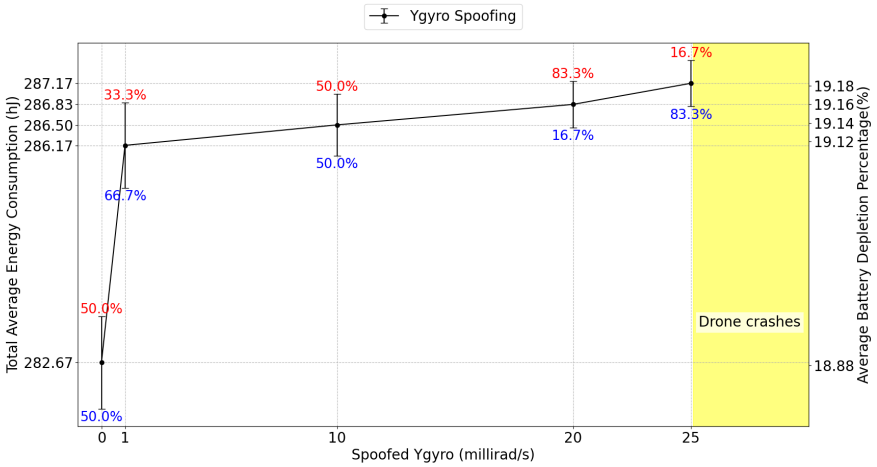


(c) Impact of spoofing the Zacc sensor values of all three available accelerometers.

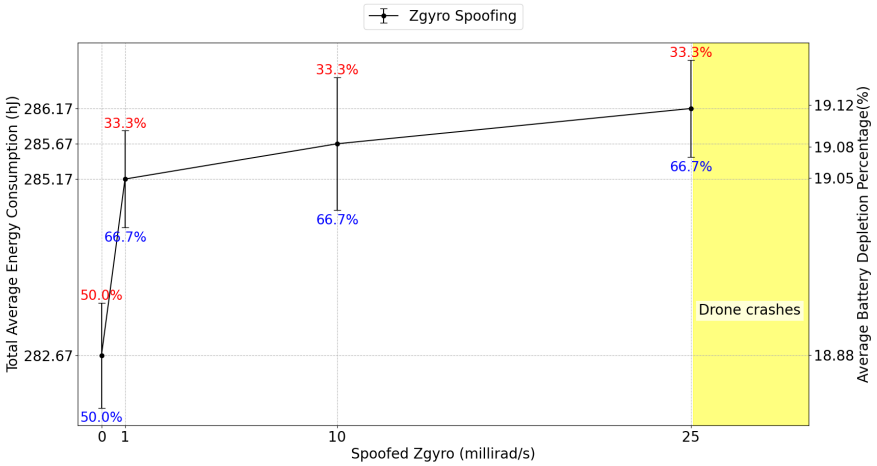
Figure 6.5: Impact of spoofing accelerometer sensor values (Xacc, Yacc, Zacc) on both total average energy consumption and its corresponding average battery depletion percentage.



(a) Impact of spoofing the Xgyro sensor values of all three available gyroscopes.



(b) Impact of spoofing the Ygyro sensor values of all three available gyroscopes.



(c) Impact of spoofing the Zgyro sensor values of all three available gyroscopes.

Figure 6.6: Impact of spoofing gyroscope sensor values (Xgyro, Ygyro, Zgyro) on total average energy consumption and its corresponding average battery depletion percentage.

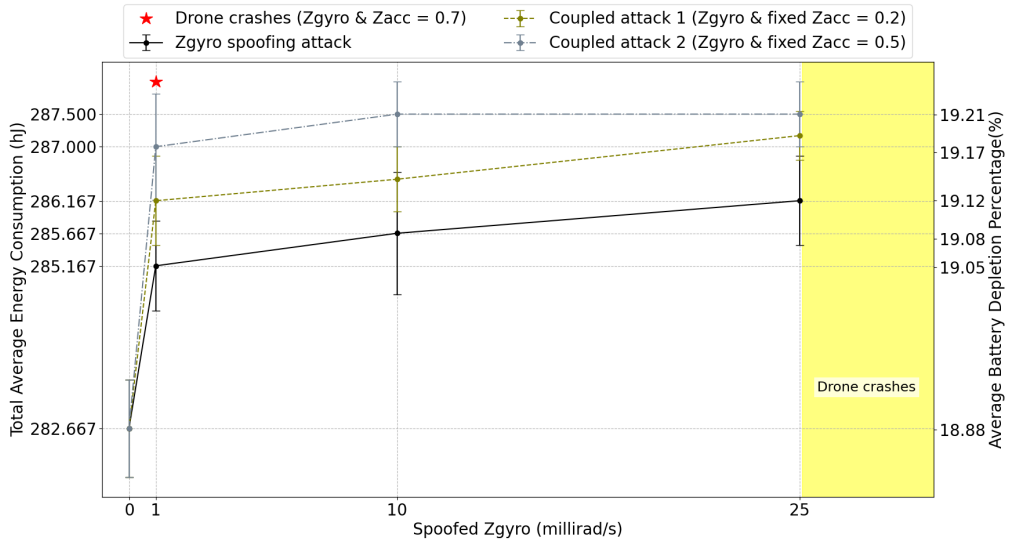


Figure 6.7: Impact of coupling attacks on both total average energy consumption of a simulated quadcopter and its corresponding average battery depletion for 1-min attack experiment.

6.5.2 Full-battery Attack Experimental Results

In this subsection, we investigate the impact of attack types tested for our full-battery experiment category. We evaluate the performance of attack types based on their observed average exhaustion time in seconds (s). As discussed previously, the exhaustion time is calculated in our experiments when the battery level of the simulated quadcopter reaches 5%. In other words, the total energy consumption of the battery spending 95% of its total capacity is 1422 hJ.

Figure 6.8 compares the results of five attack types executed for each specific attack intensity until battery exhaustion. Note that we exclude the EMI attack on the mag sensor since this attack does not have any impacts on the simulated quadcopter for 1-min experiments, using the default SITL configuration parameters. It should also be noted that when the quadcopter is not under attack and continues its normal flight condition (i.e., attack intensity is 0), the average exhaustion time is 378.9 s (see Fig. 6.8).

As seen in Fig.6.8, the higher the intensity of each attack, the lower the average battery exhaustion time. The lowest average exhaustion time (174.7 s) is observed when we run the FAI attack with its maximum attack intensity (1), showing more than 200 s decrease in average exhaustion time. However, other attacks have a similar limited impact on decreasing the average exhaustion time. Note that the red, cyan, purple, and green star signs denote the attack intensities of optical flow spoofing (0.38), GPS spoofing (0.875), EMI on SPI (0.9), and acoustic noise injection (0.9), respectively, in which the drone crashes before battery exhaustion.

Based on the observed results for average exhaustion time in Fig.6.8, we can describe the attacks except the FAI attack as *weak attacks* while the FAI attack is called a *strong attack*. To observe in more detail how weak attacks can affect the average exhaustion time, we present the results in Figure 6.9. We noticed less than 2 s decrease in the average exhaustion time when running each weak attacks on the simulated quadcopter.

Combination of Attacks.

We also consider performing the combination of two full-battery attack experiments, namely a coupled attack. This type of attack can be classified into two subclasses: 1) *Weak coupled* attacks, which consist of two weak full-battery experiments (e.g., Acoustic injection and GPS spoofing), and 2) *Strong coupled* attacks, which include a combination of one strong full-battery attack

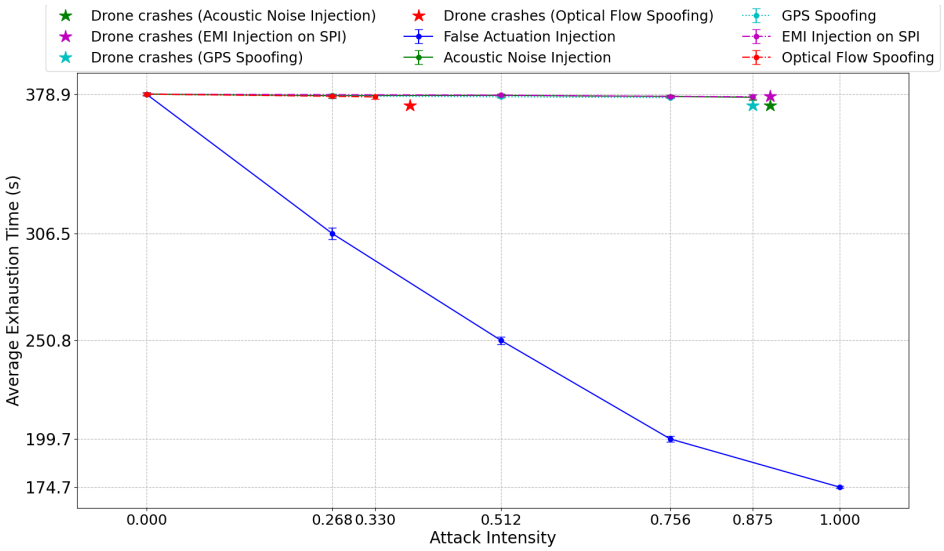


Figure 6.8: Overall comparison of attack types tested for full-battery attack experiments.

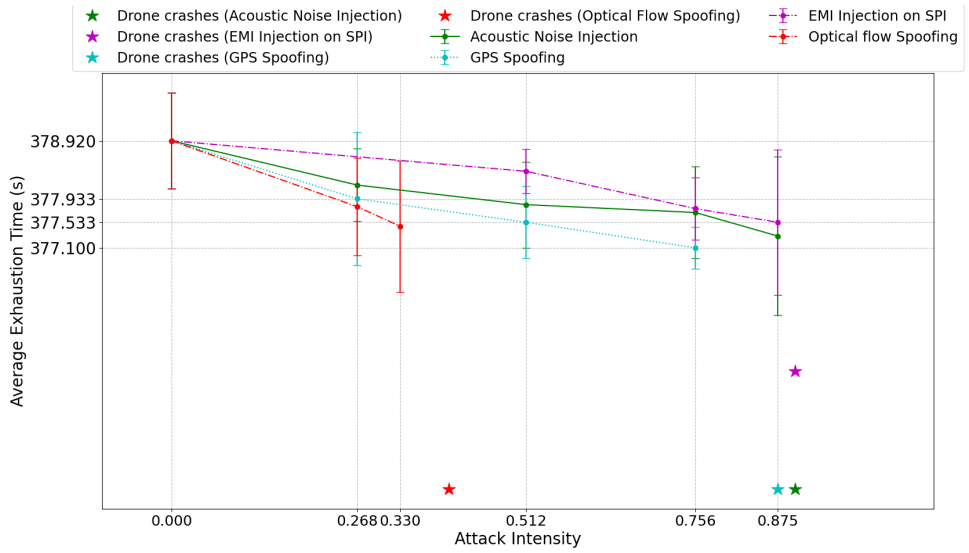


Figure 6.9: A detailed comparison of weak attacks tested for full-battery attack experiments.

experiment (i.e., FAI attack) with a weak attack experiment (e.g., GPS spoofing). Figures 6.10 and 6.11 show the results of full-battery experiments investigating the impact of a weak coupled attack (Acoustic noise injection and GPS spoofing) and a strong coupled attack (FAI attack and GPS spoofing), respectively. As seen in Fig.6.10, we observed that the combination of two weak attacks results in a weak attack with a lower average exhaustion time. For example, when we fixed the GPS spoofing attack intensity to 0.765 and varied the intensity of acoustic noise injection, the average exhaustion time for this coupled attack decreases to approximately 376.5 s, lower than the results for each one of the single attack types in Fig 6.9. Note that the attack intensity points at which the simulated drone crashes before the battery exhaustion, for each one of the three weak coupled attacks in Fig.6.10 are depicted as the brown, dark blue, and light green star signs.

Analyzing the examples of strong coupled attack experiments in Fig.6.11 shows that all the combinations of a strong attack (the FAI attack) with a weak attack (GPS spoofing) lead to a

strong coupled attack.We also observed that a decrease in average exhaustion time for a strong coupled attack occurs only if the GPS spoofing attack intensity is less than the one for FAI attack (see the coupled attack depicted by the yellow line in Fig.6.11). Specifically, the observed results for the average exhaustion time of this coupled attack are shown on the right side of the figure. It should also be noted that choosing high GPS spoofing attack intensity (i.e., 0.756) causes the drone to crash if the FAI attack intensity is 1 (see the dark brown star sign in the figure).

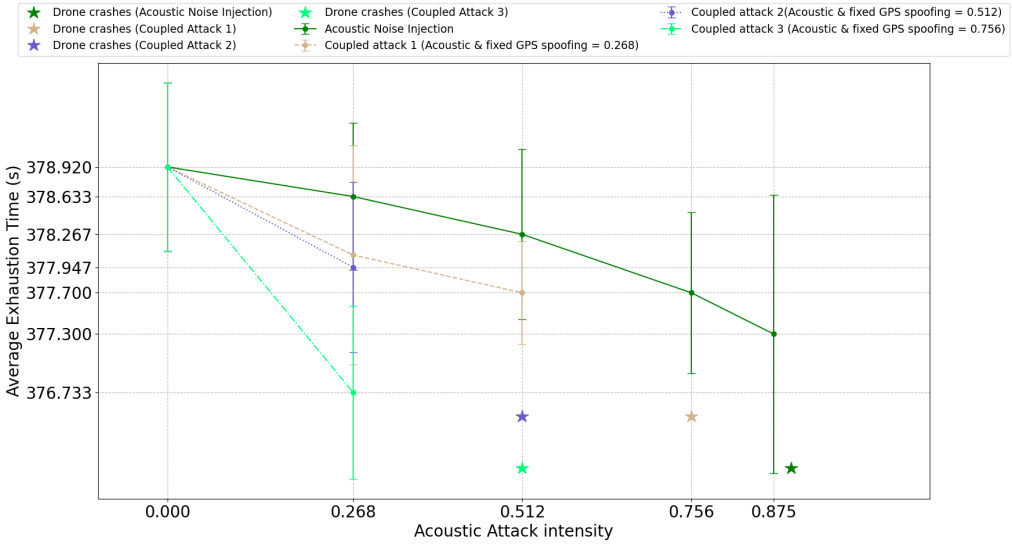


Figure 6.10: Impact of weak coupled attacks on average exhaustion time of a simulated quadcopter for full-battery experiment.

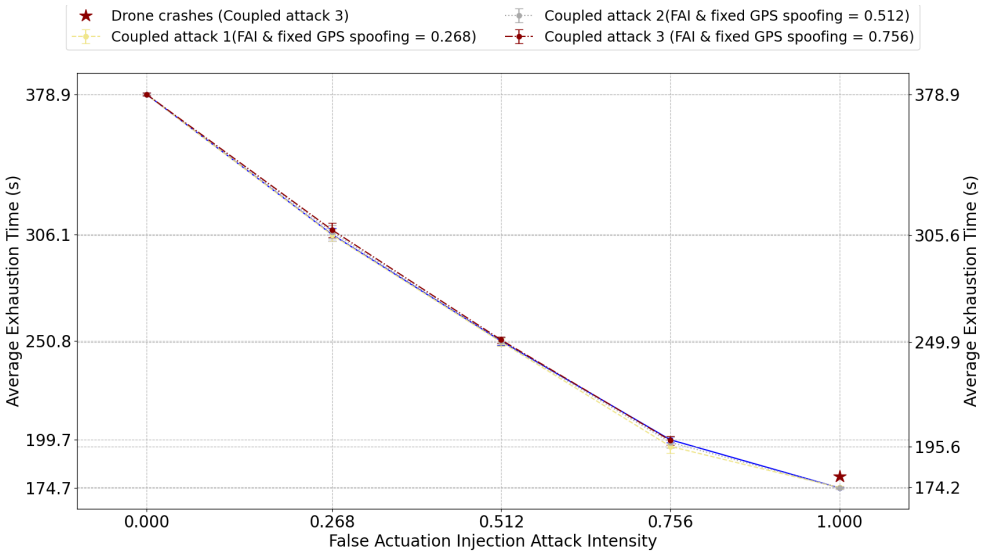


Figure 6.11: Impact of strong coupled attacks on average exhaustion time of a simulated quadcopter for full-battery experiment.

Key Takeaways.

To sum up, the most successful attack type in our full-battery experiments is the FAI attack (described as a strong attack), resulting in the lowest average exhaustion time. However, the observed results for other attack types executed in full-battery experiments are very similar and show a limited impact. These other attack types are called weak attacks. We also conclude that the combination of two weak attacks is still a weak attack, with a slight improvement in decreasing the average exhaustion time. On the other hand, a mixture of a strong attack and a weak attack leads to a strong attack, which might be more effective than the single FAI attack if the intensity of the weak attack is lower than the one for the strong attack.

Chapter 7

Conclusions

7.1 Conclusion

In this report, we have focused on modeling the impact of Physical Sensor Attacks (PSAs) and the False Actuation Injection (FAI) attack on the energy consumption of a target energy-constrained Internet of Things (IoT) device (e.g., Unmanned Aerial Vehicles (UAVs)) in ArduPilot software-in-the-loop (SITL) environment. We have introduced a probabilistic framework attack algorithm for exhausting the battery of an UAV, using a list of physical-layer attacks which are prioritized based on their direct or indirect impacts on rotating the servo motors of the target drone. We have validated our proposed attack framework through two different attack experiments (1-min and full-battery). Evaluation of 1-min and full-battery attack experiments demonstrated, respectively, over 16% increase in average battery depletion, and 200 s decrease in average exhaustion time when the attacker utilized the FAI attack with its maximum attack intensity compared to the results observed from the simulated drone, which is flying under no attack. The results obtained from testing other attacks (i.e., PSAs) showed more limited impact on the given performance metrics.

To address the main research question:

Can existing PSAs and FAI attacks be used for conducting the energy depletion attack?

We also have investigated the following sub-questions identified in Section 1.2 throughout this study, as can be seen in Table 7.1.

Question	Answered	Location
Q1. What specific sensor/actuator packets can affect and maximize the energy consumption in Low Power Wireless (LPW) devices and give a ranking to each one of these sensors?	✓	Chapter 3
Q2. How can the adversary utilize physical-layer attacks to conduct the energy depletion attacks?	✓	Section 4.2
Q3. What are the attacker's requirements and their strategies for successfully conducting the battery exhaustion attack?	✓	Chapter 5
Q4. What effect does this attack have on total energy consumption and battery depletion percentage within 1 minute?	✓	Section 6.5.1
Q5. What effect does this attack have on average exhaustion time?	✓	Section 6.5.2

Table 7.1: Research sub-questions and their answers.

To answer Q1, we came up with Table 3.3 in Chapter 3 that prioritize each attack target (i.e., sensors, a servo motor, and a GPS receiver) inside a general drone model based on their direct or indirect effect on the activation of UAV's servo motors, which consume the highest power consumption in the UAV. Regarding Q2, we introduced the adversary model in Section 4.2, which explains the objective of the attack, the capabilities of the adversary, the tools that the adversary can use to achieve its objective, and any other attack assumptions. In addition, we answered Q3 in Chapter 5 by first specifying formally the attacker's requirements in Table 5.1. We then presented in Table 5.2 how we can meet the identified requirements by proposing attack strategies. We answered both Q4 and Q5 based on our findings provided in Section 6.5.1 and Section 6.5.2, respectively. To be more specific, the observed results showed that the FAI attack is the most successful attack, causing the target UAV to consume more energy, depleting its battery more quickly within 1 minute, resulting in the lowest exhaustion time.

7.2 Limitations and future work

The most important limitation in our work is to conduct and validate the proposed attack algorithm in a real-world setting. In fact, we can extend this work by using a real drone (e.g., 3DR Iris and Solo) targeted by both PSAs and the FAI attack in practice with equipment listed in Table 4.1. Alternatively, further research can include hardware-in-the-loop (HITL) simulations, which are widely used for evaluating the flight controller software (ArduPilot and PX4) using a physical board, such as Pixhawk 1 and Pixhawk 4 Mini boards.

In this work, we also considered the impact of GPS spoofing on energy-constrained IoT devices. For future work, we can simulate a general GNSS Spoofing attack by 1) integrating the ArduPilot SITL with the Gazebo Plugin and 2) changing the default SITL configuration parameters to be able to simulate other types of GNSS signals. Applying these two considerations can help us analyze the effect of GNSS Spoofing on the energy consumption level of a simulated drone.

Also, we conducted the experiment attacks with the assumption that the target drone is in Guided mode while hovering at 3 m of height. We can extend our analysis in the future by considering different drones' flight status modes (e.g., moving from one point A to another point B during a mission).

We also do not consider how spoofing barometer sensors can impact the energy of the target quadcopter in SITL. We leave this topic for future work.

Finally, we can add other environmental conditions into our experiments, rather than the wind speed and direction, into ArduPilot SITL by integrating this simulation platform with the Gazebo simulator. This integration can enable us to take other weather conditions, such as the temperature, into account since this environmental factor can have a direct impact on the energy consumption of a victim drone.

Bibliography

[1] 3D Robotics (3DR). https://www.drones.nl/drones/3d-robotics-solo/manual, 2016. 23, 26

[2] Hasini Viranga Abeywickrama, Beeshanga Abewardana Jayawickrama, Ying He, and Eryk Dutkiewicz. Empirical power consumption model for uavs. In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pages 1–5. IEEE, 2018. 8, 19, 21

[3] Anum Ali, Ghalib A Shah, and Junaid Arshad. Energy efficient techniques for m2m communication: A survey. *Journal of Network and Computer Applications*, 68:42–55, 2016. 6

[4] Irina Artemova. https://windy.app/blog/wind-speed-beaufort-scale.html, 2025. 21

[5] Gonglong Chen and Wei Dong. Reactive jamming and attack mitigation over cross-technology communication links. *ACM Transactions on Sensor Networks (TOSN)*, 17(1):1–25, 2020. 5, 6

[6] Pritam Dash, Mehdi Karimibiuki, and Karthik Pattabiraman. Out of control: stealthy attacks against robotic vehicles protected by control-based techniques. In *Proceedings of the 35th Annual Computer Security Applications Conference*, pages 660–672, 2019. 19

[7] Drew Davidson, Hao Wu, Rob Jellinek, Vikas Singh, and Thomas Ristenpart. Controlling UAVs with sensor input spoofing attacks. In *10th USENIX workshop on offensive technologies (WOOT 16)*, 2016. 6, 7

[8] Gökçen Yılmaz Dayanıklı, Sourav Sinha, Devaprakash Muniraj, Ryan M Gerdes, Mazen Farhood, and Mani Mina. Physical-Layer attacks against pulse width Modulation-Controlled actuators. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 953–970, 2022. 1, 4, 6, 7, 11, 19, 24

[9] Vasily Desnitsky and Igor Kotenko. Simulation and assessment of battery depletion attacks on unmanned aerial vehicles for crisis management infrastructures. *Simulation Modelling Practice and Theory*, 107:102244, 2021. 1, 5, 6

[10] Karel Domin, Iraklis Symeonidis, and Eduard Marin. Security analysis of the drone communication protocol: Fuzzing the mavlink protocol. *Symposium on Information Theory*, 2016. 5, 6

[11] Jake Dulligan, Laura Freeman, Austin Phoenix, and Bradley Davis. The rising threat of non-state actor commercial drone use: Emerging capabilities and threats. 2025. 17

[12] N Geethanjali and E Gayathri. A survey on energy depletion attacks in wireless sensor networks. *International Journal of Science and Research*, 3(9):2070–2074, 2014. 6

[13] Hao Gong, Baoqi Huang, Bing Jia, and Hansu Dai. Modelling power consumptions for multi-rotor uavs. *IEEE Transactions on Aerospace and Electronic Systems*, 2023. 8, 10

[14] Kim Hartmann and Christoph Steup. The vulnerability of uavs to cyber attacks-an approach to the risk assessment. In *2013 5th international conference on cyber conflict (CYCON 2013)*, pages 1–23. IEEE, 2013. 6

[15] Jason Baker (Red Hat). https://opensource.com/article/18/2/drone-projects, 2018. 19

[16] Zhitao He and Thiemo Voigt. Droplet: A new denial-of-service attack on low power wireless sensor networks. In *2013 IEEE 10th International Conference on Mobile Ad-Hoc and Sensor Systems*, pages 542–550. IEEE, 2013. 6

[17] Folker Hoffmann, Matthew Ritchie, Francesco Fioranelli, Alexander Charlish, and Hugh Griffiths. Micro-doppler based detection and tracking of uavs with multistatic radar. In *2016 IEEE radar conference (RadarConf)*, pages 1–6. IEEE, 2016. 10, 14, 17

[18] Hongning Hou, Jian Zhuang, Hu Xia, Guanwei Wang, and Dehong Yu. A simple controller of minisize quad-rotor vehicle. In *2010 IEEE International Conference on Mechatronics and Automation*, pages 1701–1706, 2010. 19

[19] Joon-Ha Jang, Mangi Cho, Jaehoon Kim, Dongkwan Kim, and Yongdae Kim. Paralyzing drones via emi signal injection on sensory communication channels. In *NDSS*, 2023. ix, 4, 6, 7, 11

[20] Jinseob Jeong, Dongkwan Kim, Joon-Ha Jang, Juhwan Noh, Changhun Song, and Yongdae Kim. Un-rocking drones: Foundations of acoustic injection attacks and recovery thereof. In *NDSS*, 2023. 6, 7

[21] Hyungsub Kim, Rwitam Bandyopadhyay, Muslum Ozgur Ozmen, Z Berkay Celik, Antonio Bianchi, Yongdae Kim, and Dongyan Xu. A systematic study of physical sensor attack hardness. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 143–143. IEEE Computer Society, 2024. xi, 1, 4, 6, 7, 11, 20, 21, 24, 25, 28, 29

[22] Harini Kolamunna, Thilini Dahanayaka, Junye Li, Suranga Seneviratne, Kanchana Thilakaratne, Albert Y Zomaya, and Aruna Seneviratne. Droneprint: Acoustic signatures for open-set drone detection and identification with online data. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(1):1–31, 2021. 10, 14, 17

[23] Anthony Bahadir Lopez, Korosh Vatanparvar, Atul Prasad Deb Nath, Shuo Yang, Swarup Bhunia, and Mohammad Abdullah Al Faruque. A security perspective on battery systems of the internet of things. *Journal of Hardware and Systems Security*, 1:188–199, 2017. 5, 6

[24] Ian A Mason, Vivek Nigam, Carolyn Talcott, and Alisson Brito. A framework for analyzing adaptive autonomous aerial vehicles. In *International Conference on Software Engineering and Formal Methods*, pages 406–422. Springer, 2017. 23, 26

[25] Lorenz Meier. https://mavlink.io/en/, 2025. 24

[26] Yassine Mekdad, Ahmet Aris, Leonardo Babun, Abdeslam El Fergougui, Mauro Conti, Riccardo Lazzeretti, and A Selcuk Uluagac. A survey on security and privacy issues of uavs. *Computer networks*, 224:109626, 2023. 3

[27] Aristides Mpitziopoulos, Damianos Gavalas, Charalampos Konstantopoulos, and Grammati Pantziou. A survey on jamming attacks and countermeasures in wsns. *IEEE communications surveys & tutorials*, 11(4):42–56, 2009. 6

[28] Van-Linh Nguyen, Po-Ching Lin, and Ren-Hung Hwang. Energy depletion attacks in low power wireless networks. *IEEE Access*, 7:51915–51932, 2019. 5, 6

[29] Hossein Pirayesh and Huacheng Zeng. Jamming attacks and anti-jamming strategies in wireless networks: A comprehensive survey. *IEEE communications surveys & tutorials*, 24(2):767–809, 2022. 5, 6

[30] Matthew Pirretti, Sencun Zhu, Narayanan Vijaykrishnan, Patrick McDaniel, Mahmut Kandemir, and Richard Brooks. The sleep deprivation attack in sensor networks: Analysis and methods of defense. *International Journal of Distributed Sensor Networks*, 2(3):267–287, 2006. 6

[31] David R Raymond and Scott F Midkiff. Clustered adaptive rate limiting: Defeating denial-of-sleep attacks in wireless sensor networks. In *MILCOM 2007-IEEE military communications conference*, pages 1–7. IEEE, 2007. 6

[32] Renogy. https://www.renogy.com/blog/what-is-depth-of-discharge-for-battery, 2025. 26

[33] Michael Robinson. Knocking my neighbors kids cruddy drone offline. *DEF CON*, 23, 2015. 6, 7

[34] Harshad Sathaye, Martin Strohmeier, Vincent Lenders, and Aanjhan Ranganathan. An experimental study of GPS spoofing and takeover attacks on UAVs. In *31st USENIX security symposium (USENIX security 22)*, pages 3503–3520, 2022. 6, 7

[35] Savio Sciancalepore, Omar Adel Ibrahim, Gabriele Oligeri, and Roberto Di Pietro. Pinch: An effective, efficient, and robust solution to drone detection via network traffic analysis. *Computer Networks*, 168:107044, 2020. 10, 14, 17

[36] Econocom Digital Security. https://github.com/DigitalSecurity/btlejuice, 2018. 5

[37] Jayaprakash Selvaraj, Gökçen Yılmaz Dayanıklı, Neelam Prabhu Gaunkar, David Ware, Ryan M Gerdes, and Mani Mina. Electromagnetic induction attacks against embedded systems. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, pages 499–510, 2018. 6, 7, 11, 24

[38] Muhammad Ali Siddiqi, Wouter A Serdijn, and Christos Strydis. Zero-power defense done right: Shielding imds from battery-depletion attacks. *Journal of Signal Processing Systems*, 93:421–437, 2021. 6

[39] Leydson Silva, Ewerton Salvador, Alisson Brito, Jose Sousa Barros, and Vivek Nigam. A multi-uav co-simulation environment for safety and performance analysis. In *2019 IX Brazilian Symposium on Computing Systems Engineering (SBESC)*, pages 1–8. IEEE, 2019. 23, 26

[40] Dennis Silverman. https://www.physics.uci.edu/~silverma/units.html, 2025. 26

[41] Yunmok Son, Hocheol Shin, Dongkwan Kim, Youngseok Park, Juhwan Noh, Kibum Choi, Jungwoo Choi, and Yongdae Kim. Rocking drones with intentional sound noise on gyroscopic sensors. In *24th USENIX security symposium (USENIX Security 15)*, pages 881–896, 2015. ix, 3, 6, 7, 10, 16

[42] Ardupilot Dev Team. https://github.com/ArduPilot/ardupilot/blob/master/BUILD.md, 2016. 19

[43] Ardupilot Dev Team. https://ardupilot.org/, 2024. 19

[44] Ardupilot Dev Team. https://ardupilot.org/copter/, 2024. 19

[45] Ardupilot Dev Team. https://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html, 2024. 19

[46] Ardupilot Dev Team. https://ardupilot.org/copter/docs/what-is-a-multicopter-and-how-does-it-work.html#what-is-a-multicopter-and-how-does-it-work, 2024. 19

[47] ArduPilot Dev Team. https://ardupilot.org/plane/docs/parameters.html#parameters-sim, 2024. 19, 24

[48] Ardupilot Dev Team. https://ardupilot.org/copter/docs/common-servo.html, 2024. 19

[49] Pixhawk Development Team. https://px4.io/, 2023. 19

[50] Nils Ole Tippenhauer, Christina Pöpper, Kasper Bonne Rasmussen, and Srdjan Capkun. On the requirements for successful gps spoofing attacks. In *Proceedings of the 18th ACM conference on Computer and communications security*, pages 75–86, 2011. 7

[51] Fadhila Tlili, Lamia Chaari Fourati, Samiha Ayed, and Bassem Ouni. Investigation on vulnerabilities, threats and attacks prohibiting uavs charging and depleting uavs batteries: Assessments & countermeasures. *Ad hoc networks*, 129:102805, 2022. 1, 5, 6

[52] Timothy Trippel, Ofir Weisse, Wenyuan Xu, Peter Honeyman, and Kevin Fu. Walnut: Waging doubt on the integrity of mems accelerometers with acoustic injection attacks. In *2017 IEEE European symposium on security and privacy (EuroS&P)*, pages 3–18. IEEE, 2017. 6, 7

[53] Yazhou Tu, Zhiqiang Lin, Insup Lee, and Xiali Hei. Injected and delivered: Fabricating implicit control over actuation systems by spoofing inertial sensors. In *27th USENIX security symposium (USENIX Security 18)*, pages 1545–1562, 2018. 6, 7

[54] Paparazzi UAS. https://github.com/paparazzi/paparazzi/, 2023. 19

[55] GNU Lesser General Public License v3. https://pypi.org/project/pymavlink/, 2025. 24

[56] Gabriel Vasconcelos, Gabriel Carrijo, Rodrigo Miani, Jefferson Souza, and Vitor Guizilini. The impact of dos attacks on the ar. drone 2.0. In *2016 XIII Latin American robotics symposium and IV Brazilian robotics symposium (LARS/SBR)*, pages 127–132. IEEE, 2016. 6

[57] Eugene Y Vasserman and Nicholas Hopper. Vampire attacks: Draining life from wireless ad hoc sensor networks. *IEEE transactions on mobile computing*, 12(2):318–332, 2011. 6

[58] Zhengbo Wang, Kang Wang, Bo Yang, Shangyuan Li, and Aimin Pan. Sonic gun to smart devices: Your devices lose control under ultrasound/sound. *Black Hat USA*, pages 1–50, 2017. 6, 7

[59] Anthony D Wood, John A Stankovic, and Gang Zhou. Deejam: Defeating energy-efficient jamming in ieee 802.15. 4-based wireless networks. In *2007 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, pages 60–69. IEEE, 2007. 6

[60] Joshua Wright. https://github.com/riverloopsec/killerbee, 2018. 5

[61] Wenyuan Xu, Ke Ma, Wade Trappe, and Yanyong Zhang. Jamming sensor networks: attack and defense strategies. *IEEE network*, 20(3):41–47, 2006. 6

[62] Kexiong Curtis Zeng, Shinan Liu, Yuanchao Shu, Dong Wang, Haoyu Li, Yanzhi Dou, Gang Wang, and Yaling Yang. All your GPS are belong to us: Towards stealthy manipulation of road navigation systems. In *27th USENIX security symposium (USENIX security 18)*, pages 1527–1544, 2018. 6, 7

[63] Yueyan Zhi, Zhangjie Fu, Xingming Sun, and Jingnan Yu. Security and privacy issues of uav: A survey. *Mobile Networks and Applications*, 25(1):95–101, 2020. 1, 7