A Systematic Study of Physical Sensor Attack Hardness

Hyungsub Kim
Purdue University
kim2956@purdue.edu

Rwitam Bandyopadhyay
Purdue University
bandyopr@purdue.edu

Muslum Ozgur Ozmen
Purdue University
mozmen@purdue.edu

Z. Berkay Celik
Purdue University
zcelik@purdue.edu

Antonio Bianchi
Purdue University
antoniob@purdue.edu

Yongdae Kim
KAIST
yongdaek@kaist.ac.kr

Dongyan Xu
Purdue University
dxu@purdue.edu

***Abstract*—Physical sensor attacks against robotic vehicles (RV) have become a serious concern due to their prevalence and potential physical threat. However, RV software developers often do not deploy appropriate countermeasures. This hesitance stems from their belief that attackers face substantial challenges when conducting sensor attacks, e.g., nullifying sensor redundancy in hardware and circumventing sensor filters in software. Yet, we discover that attackers can overcome the challenges by fulfilling specific prerequisites and finely tuning attack parameters. The misconceptions that the developers have arisen from a lack of study regarding the level of difficulty attackers face in successfully achieving their attack goals, which we call "attack hardness".**

In this paper, we examine the hardness of 12 well-known sensor attacks. We first identify the prerequisites required to conduct the attacks successfully. We then quantify the hardness of each attack as how frequent the prerequisites enabling a specific attack are in the real world. To automate this analysis, we introduce RVPROBER, an attack prerequisite analysis framework. RVPROBER discovered that the 12 sensor attacks require, on average, 4.4 prerequisites, highlighting that previous literature has often missed important details required to perform these attacks. By satisfying the identified prerequisites and tuning attack parameters, we increased the number of successful attacks from 6 to 11. Moreover, our analysis showed that an average of 57.08% of actual RV users are vulnerable to sensor attacks. Finally, starting from the identified prerequisites, we analyzed the reasons behind the success of each attack and found previously-unknown root causes, such as design flaws in the RV software's fail-safe logic.

1. Introduction

Attacks affecting robotic vehicles (RVs) have become a serious concern in recent years due to their prevalence and their potentially-severe consequences (ranging from destroying physical infrastructure to causing human injuries). Many of these attacks fall into the category of "physical sensor attacks", which focus on tampering with the mechanisms an RV uses to perceive its operating environment [22], [42], [43], [82], [85], [86], [88], [89], [93], [97]. For instance, GPS spoofing [83], [86], [97] aims to deceive an RV's GPS receiver by sending fake GPS signals, potentially causing it to lose position control and crash, resulting in severe damage.

Unfortunately, RV control software remains vulnerable to physical sensor attacks. In fact, a preliminary analysis we performed on the most popular open-source RV software (ArduPilot [4], PX4 [66], and Paparazzi [64]), discovered that RV software developers do not deploy any countermeasures intended to prevent this kind of attacks.

Additionally, throughout our communication with RV software developers (detailed in Section 2), we observed a lack of motivation to defeat sensor attacks. In fact, they believe that these attacks are not practical since they require attackers to overcome hard challenges, such as nullifying sensor redundancy in hardware and circumventing sensor filters in software. Thus, they hold the view that it is hard for the attackers to replicate the attacks in real world scenarios.

Indeed, attempting to replicate these attacks is challenging, and in a preliminary study we also failed to replicate six out of 12 well-known sensor attacks. However, as we will show, a more in-depth analysis of these attacks reveals that the attackers can use these attacks in real-world scenarios by fulfilling specific attack prerequisites and finely tuning attack parameters. Particularly, our investigation reveals that successful sensor attacks require prerequisites in hardware, software, and environmental configuration categories. Additionally, we find that sensor attacks do not function on a "plug-and-play" basis; instead, they demand meticulous tuning of attack parameters, such as the intensity and duration of spoofed signals, to achieve attack goals.

Fundamentally, the misunderstanding that the developers have arises from a lack of "attack hardness" study. Attack hardness refers to a quantification of the difficulty involved in executing an attack. Specifically, for a given sensor attack, it quantifies the attack hardness as the likelihood that these prerequisites are met among a representative sample of real-world RV usage scenarios.

Identifying prerequisites and attack hardness enables developers to successfully replicate the attacks, identify the RV configurations vulnerable to such attacks, and understand the attack impact and prevalence. In turn, these capabilities allow

developers to develop countermeasures (i.e., code patches and/or hardware modifications) and test them, prioritizing fixes for the most impactful attack. However, current research literature [22], [42], [43], [82], [83] does not provide a complete list of required prerequisites, and there currently exists no automated tool that identifies such prerequisites and leverages them to quantify the attack hardness.

In this paper, motivated by the lack of effective approaches to understanding the hardness of attacks, we study the hardness of 12 well-known physical sensor attacks. In particular, we first identify the prerequisites required to conduct the attacks successfully. We then quantify the hardness of each attack as how frequent the prerequisites enabling a specific attack are in the real world.

To automate this process, we develop RVPROBER, an attack prerequisite analysis framework. To simulate realistic sensor attacks while considering various attack parameters, RVPROBER takes, as input, sensor values compromised under the attack. Then, it mutates the sensor values based on four different attack parameters (intensity, duration, start time, and attack algorithm). To address the high-dimensional "search spaces" involving the attack prerequisites, RVPROBER reduces the spaces via dedicated static and dynamic analyses.

Using RVPROBER, we discover that the 12 physical sensor attacks require, on average, 4.4 different types of prerequisites, many of them not reported in the research papers presenting them. We share the identified prerequisites with the authors of research papers proposing 12 physical sensor attacks [22], [42], [43], [82], [83]. Thus, we confirm that our findings are generally in line with the authors' observations. Additionally, by ensuring the identified prerequisites and attack parameters are met, we increase the number of successful attacks from 6 to 11. To quantify the real-world hardness of conducting the attacks, we verify whether the required prerequisites are satisfied by the conditions and the configurations of RVs operated in real-world scenarios. To do so, we collect a total of 30,059 real RV logs from actual users of two popular RV control software (ArduPilot [4] and PX4 [66]). As a result, we discover that physical sensor attacks are possible, on average, in 57.08% of real-world RV configurations.

Finally, we use our ability to successfully replicate attacks to study in-depth their root causes. This study reveals previously-unknown root causes of successful attacks. In particular, we found four of these attacks are possible due to design flaws in the RV control software's fail-safe logic.

Overall, our findings shed light on the feasibility (or unfeasibility) of physical sensor attacks against RVs, and they will help current and future RV control software developers to better understand these attacks, allowing them to introduce, when needed, effective countermeasures against them.

In summary, we make the following contributions:

- **Discovery of Attack Difficulty.** Our research highlights the difficulty of conducting and studying sensor attacks due to the prerequisites required for successfully conducting the attacks.
- **Attack Hardness Analysis Tool.** RVPROBER conducts realistic sensor attacks using simulations. Each simulation tests distinct prerequisites. Based on the prerequisites found by RVPROBER, we increase the number of successful attacks from 6 to 11. The identified prerequisites not only help developers understand the impact and prevalence of these attacks, but also provide the insights needed to devise and test countermeasures.
- **Evaluation with Real-world Data.** We analyze how frequently RV configurations used by actual users satisfy the required prerequisites by analyzing a total of 30,059 RV logs. Our analysis shows that an average of 57.08% of actual RV users are vulnerable to the attacks.
- **Root Cause Analysis.** By analyzing their prerequisites, we identify the root causes enabling the successful attacks. This analysis reveals that many attacks are possible due to previously-unknown design flaws in RV software's fail-safe algorithms.

We make RVPROBER and experimental data publicly available (https://github.com/purseclab/RVProber).

Ethical Considerations and Responsible Disclosure. In the analysis of public RV logs of actual users, we analyze the publicly available RV logs and do not collect any personally identifiable information (PII). Additionally, we responsibly disclosed the physical sensor attacks and the discovered design flaws to the corresponding RV software developers.

2. Background

Sensor Fusion. Each RV control software can leverage a different sensor fusion algorithm. Here, we explain sensor fusion commonly used in popular open-source RV control software (ArduPilot [4], PX4 [66], and Paparazzi [64]). Sensor fusion aims to improve the accuracy and reliability of estimating an RV's physical states (e.g., position). It comprises three primary components: sensors, sensor filters, and sensor fusion algorithms (e.g., EKFs), as shown in Figure 1. Most RVs are equipped with redundant homogeneous and heterogeneous sensors to address the randomness of sensor readings (e.g., disturbances and failures). For example, an RV is equipped with three redundant gyroscopes, three accelerometers, and two magnetometers to measure the RV's attitude (roll, pitch, and yaw angles), angular velocity, and position (❶ in Figure 1). The two GNSS receivers can also measure the RV's attitude (yaw angles) [35] and are not the only sensors that measure the RV's positions. In fact, RV control software leverages all sensors (except for the barometer and range finder) to determine the RV's latitude and longitude coordinates. RV control software next inputs the measured sensor values into the sensor filters, e.g., low-pass and harmonic notch filters [11], [76] (❷). Lastly, RV control software simultaneously feeds the filtered sensor values to multiple extended Kalman filters (EKFs) (❸).

The number of EKFs is equal to the number of redundant IMU sensors (three gyroscopes and accelerometers in Figure 1). Each EKF takes, as input, a different instance of the

Figure 1: Illustration of different types of sensors (❶), sensor filters (❷), and sensor fusion processes on RVs (❸).

TABLE 1: Illustration of the EKF switching [23]

	EKF1	EKF2	EKF3
Gyroscope (Gyro)	Gyro 1	Gyro 2	Gyro 3
Accelerometer (Acc)	Acc 1	Acc 2	Acc 3
Magnetometer (Mag)	Mag 1	Mag 2	Mag 1
Barometer (Bar)	Bar 1	Bar 2	Bar 1
Range finder (Ran)	Ran 1	Ran 1	Ran 1
GNSS	GNSS 1	GNSS 2	GNSS 1
Optical flow (Opt)	Opt 1	Opt 1	Opt 1

sensor, as shown in Table 1. For example, EKF2 uses the second instance of a gyroscope, accelerometer, magnetometer, barometer, and GNSS, and the first instance of a range finder and optical flow sensors. Each EKF then estimates the RV's current physical states separately. RV software uses only one of the EKFs' outputs rather than combining them.

EKF Switching. This sensor fusion design allows excluding a sensor that returns erroneous measurements. Particularly, the RV control software calculates each EKF's error score by considering differences between the EKF's estimation and the actual sensor measurement. It then switches the primary EKF to the secondary EKF (i.e., EKF1 to EKF2) if the primary EKF shows the poorest error score among EKFs, i.e., the RV leverages the second instance of the gyroscope sensor rather than the first and third ones [23].

Attacks. Physical sensor attacks remotely inject/jam signals to disrupt sensors attached to RVs, as shown in Table 2. Throughout the paper, we consistently use the term "attacks" to refer to these physical sensor attacks.

GNSS receivers[1] are vulnerable to spoofing and jamming attacks because (i) the strength of legitimate GNSS signals is extremely weak (-125 dBm to -130 dBm in open space) and (ii) GNSS receivers do not leverage authentication methods. Attackers deceive GNSS receivers by either spoofing fake signals that are stronger than legitimate ones or by jamming legitimate signals [83], [86], [97]. Such attacks may lead to unstable position control and a physical crash into an obstacle. To address natural disturbances in outdoor spaces, GNSS receivers parse and track multiple types of satellite signals instead of relying solely on one type (e.g., GPS). Further, the GNSS receivers detect drastic signal level changes (e.g., noise level [38], [41]) and return warning messages, e.g., u-blox series raise the warnings in UBX-NAV-STATUS [90], [91]. Researchers have proposed a defense method that leverages angle-of-arrival measured by multi-GNSS receivers or antenna-array [49]. Yet, this method is rarely adopted due to the required additional hardware. While Galileo supports authenticated navigation messages (Galileo's OSNMA [28]), Motallebighomi et al. show that attackers can still spoof the GNSS receivers without modifying these authenticated messages [56].

1. GNSS includes any constellation systems such as GPS, Galileo, and GLONASS. In contrast, GPS denotes only the USA's positioning system.

TABLE 2: Physical sensor attacks and goals of an adversary

Attack (S): Spoofing (J): Jamming	Target sensor	Attack's goals		
		Mission failure	Unstable attitude/position	Physical crash
(S) GNSS signals [83]	GNSS	3	3	3
(J) GNSS signals [83]		3	7	7
(S) Acoustic noises [43], [85], [88], [89], [93]	Gyroscope Accelerometer	3	3	3
(S) Magnetic fields [82]	Magnetometer	3	3	3
(S) Magnetic fields [42]	SPI/I2C buses	3	3	3
(S) Images on floors [22]	Optical-flow	3	3	3

Researchers have demonstrated that acoustic noises can produce resonance inside gyroscope and accelerometer sensors (i.e., IMU) [43], [85], [88], [89], [93]. This can lead to unstable attitude control and, eventually, a crash on the ground. In response to this attack, researchers have proposed sensor filter algorithms to be implemented either within or outside the sensors [43], [94]. Additionally, fiber-optic gyroscopes have emerged as a promising alternative due to their inherent immunity to resonance-based attacks [19], [48]. Yet, all of these defense methods are not currently adopted by the developers due to the required hardware modifications, processing overhead, and/or increased costs.

Electromagnetic interference (EMI) can disturb either magnetometers [82] or data on SPI and I2C buses [42], i.e., corrupting all sensors on an RV. Such attacks lead to physical crashes due to the corrupted sensor readings. Unfortunately, RV developers have implemented EMI filters (e.g., L-C, Pi, T filters) [44] in the hardware level for only power lines (i.e., battery) [30]. Thus, sensors and SPI/I2C buses are still vulnerable to EMI attacks. To tackle natural magnetic anomalies, some RV developers attach EMI shielding (e.g., Mu-metal) to RVs [25]. Yet, as Jang et al. discussed [42], complete shielding of flight controller boards is not feasible due to the potential for heat buildup, which can degrade the performance of circuit components.

Lastly, attackers can inject images on floors via laser beams [22], causing an RV's optical flow sensor to calculate fake motions. This attack results in unstable position control and potentially causes a crash into an obstacle. To address this attack, researchers have developed algorithms to filter out injected images [22], [29]. Despite the availability of defense algorithms against this attack, RV developers have yet to incorporate these methods into their software. This reluctance stems from the inherent limitations of existing defense algorithms, which assume that attackers can only manipulate a portion of the field of view. This limitation makes these defenses vulnerable to attacks that inject spoofed images into the entire field of view.

Reasons for Not Using Countermeasures. We could not find discussions regarding the attacks except for GNSS jamming in GitHub repositories and developer community websites [6],

TABLE 3: 12 attacks (six different types of attacks with two different levels of adversary capabilities)

Attack (S): Spoo ng (J): Jamming	Adversary Capabilities	
	Attacks demonstrated in the papers	Worst-case attacks
(S) GNSS signals [83]	GPS signals	GNSS signals
(J) GNSS signals [83]	Intermittently jam	Fully jam
(S) Acoustic noises [43], [85], [88], [89], [93]	One IMU	All IMUs
(S) Magnetic elds [82]	One mag sensor	All mag sensors
(S) Magnetic elds [42]	Partial corruption	Full corruption
(S) Images on oors [22]	Partial eld of view	Entire eld of view

[68]. Thus, we made responsible disclosure, and in this regard, we reported the 10 well-known attacks (except for two GNSS jamming attacks, as the developers already discussed) to the developers of ArduPilot [4], PX4 [66], and Paparazzi [64].

We received two distinct types of replies from the developers. Firstly, they believe that the attacks can be countered by different hardware and software con gurations (e.g., the latest ight control boards and nely tuned sensor lters). Thus, they hold the belief that the attacks do not pose genuine threats. Secondly, the developers believe that addressing the attacks falls within the purview of hardware designers rather than RV software developers, as these attacks start at the sensor hardware. Yet, we will detail that the beliefs held by the developers are misconceptions in Section 9.

3. Threat Model and Scope

Our analysis focuses on six different types of remote attacks on RVs that disrupt an RV's sensors and result in a mission failure, unstable attitude/position control, or physical crash, as shown in Table 2. Within our scope, we include attacks directed toward RVs that have been documented and reported in the research literature. We assume attackers cannot inject messages between the RV and the ground control station (GCS). This is because RVs can use network packet-level authentication and encryption according to their communication protocol and hardware speci cations [54], [55], [81]. Thus, we exclude remotely exploiting bugs that corrupt sensor values stored in the RV's memory.

Adversary's Capabilities. The majority of research literature [22], [42], [43], [82], [83] leverages two distinct levels of an adversary's capabilities: i) (attacks demonstrated in research papers and ii) (worst-case attacks. Thus, we also consider these two levels of an adversary's capabilities. Table 3 shows 12 attacks classi ed by these two capabilities.

The term “demonstrated attacks” denotes attacks that have been substantiated by conducting real-world experiments in the research literature. For instance, acoustic noise attacks demonstrated in the papers [43], [85], [88], [89], [93] inject sound at a speci c frequency and compromise one of the IMUs on an RV. To compromise all IMUs, attackers need to inject sound at multiple frequencies because each IMU may have a different resonance frequency. Yet, the acoustic signals at multiple frequencies suffer from sound wave interference (i.e., phase shifts and inversion of the signal polarity [18], [87]); thus, the demonstrated attacks do not simultaneously disturb multiple IMUs.

Figure 2: (a) The location where we performed GPS spoo ng. The green line represents how much the location deviates from the correct position (in the center of the circle).

On the contrary, the term “worst-case attacks” denotes the most powerful adversary's capabilities, which are not veri ed by conducting real-world experiments due to their hardware and software limitations. We assume that the worst-case attacks overcome the challenges of conducting attacks. Thus, such attacks can simultaneously and completely compromise all of the homogeneous sensors. For instance, the worst-case attacks inject sound at multiple frequencies at the same time and succeed in disturbing all IMUs of an RV.

Adversary's Knowledge. We consider two distinct adversary knowledge models: black-box and white-box attacks. Firstly, the black-box attacks lack knowledge about a target RV's software, hardware, and environmental con gurations. Secondly, the white-box attacks have such knowledge about the target RV before they conduct attacks. Thus, the white-box attacks can dynamically select an optimal attack from the 12 available attacks and their associated parameters, thereby increasing the likelihood of achieving the attack goals.

4. Motivating Example

We demonstrate the importance of understanding attack hardness by showcasing GPS spoo ng that aims to control RVs. To conduct this attack, we follow the attack steps in [97] for RVs in an outdoor environment (See Appendix B for details). This attack only requires affordable commodity hardware, while open-source software is readily available to execute the required attack steps. To address safety and ethical concerns, we ensure that there were no man-made structures, vehicles, and people within 30 meters around the experiment location, as shown in Figure 2-(a), and we used 10 dBm low-power signals (so that the signal can reach, at most, targets within a 15-meter radius).

We fail to fully observe the reported effects of the GPS spoo ng attack. As shown in Figure 2-(b), the location perceived by the RV does not deviate from the real RV's location while performing the attack.

Reasons for Failure. With further investigation, we realized there were two main reasons why the attack failed. In other words, this attack requires two prerequisites. First, the attack is only possible under an RV's speci c “sensor con guration”. Speci cally, this attack is only possible against RVs using a GPS receiver instead of a GNSS receiver. Unlike GPS receivers, GNSS receivers use data from multiple satellite constellations (e.g., GPS, Galileo, GLONASS, BeiDou, and SBAS). For this reason, the attempted attack, which only

Figure 3: A simplified attack-step model illustrates the propagation of acoustic noise injection attacks within RVs.

affects the signal coming from the GPS satellite constellation, is not possible[2]. Second, the RV control software leverages sensor fusion algorithms, as explained in Section 2. Even though the GNSS receiver incorrectly measures its positions (i.e., measuring false velocity values), the RV's IMUs still correctly estimate the RV's current velocity.

Attack Based on the Identified Attack Prerequisites. Taking into account the discovered prerequisites, we change our experiment setup as follows: 1) we replace the u-blox M8N receiver with the u-blox M6 receiver, because the u-blox M6 receiver parses only GPS signals; and 2) we change PX4's configuration parameters (EKF2_AID_MASK, CAL_GYRO_PRIO, and CAL_ACC_PRIO) to decrease the weight given to data coming from IMU sensors and increase the weight of GPS data. After these modifications, we successfully deceive the positions estimated by the RV, as shown in Figure 2-(c).

Quantified Real-world Hardness. We can use the knowledge about the required prerequisites to estimate how frequent they are to be true in real-world scenarios. To do so, we collect 30,059 real RV logs from actual users of two popular RV software [4], [66]. We discover that in 18,271 out of the 30,059 RV logs (60.8%), these two prerequisites are true.

As we will show in Section 9, RVPROBER helps us identify the required prerequisites for attacks. In turn, the identified prerequisites to understand how likely an attack is (consulting real-world RV logs, as explained) and conducting the attack help developers test their software and patch it.

5. Attack Prerequisites

We first introduce our methodology to identify attack prerequisites. Then, we provide an in-depth explanation of the prerequisites that are required for successful attacks.

Identification of Attack Prerequisites. To discern components of RV hardware and software that amplify or mitigate the impacts of the 12 considered attacks, we leverage a single "attack-step model" using hierarchical design modeling (HdM) [31], as shown in Figure 3. Each component is recursively decomposed into subsystems (Figure 3-❶). With this model, we abstract the paths through which an attack propagates across both cyber and physical components and its interactions with the environment.

The attack-step model exhibits generality across the most popular RV hardware and software platforms. Specifically, identifying hardware components within the attack-step model involves manually analyzing the most widely used open-standard flight control boards (Pixhawk series [65]). Likewise, determining software components in the attack-step model entails manually examining the documentation and source code of the most popular open-source RV software (ArduPilot, PX4, and Paparazzi). We then extract only the common hardware and software components from the flight control boards and RV software packages.

To illustrate, Figure 3 shows a simplified attack-step model to understand the propagation of acoustic noise injection attacks. A series of sensor filters and sensor fusions mitigate deviated sensor measurements (❶-❷), but they still allow RV software to incorrectly estimate physical states of RVs, e.g., roll, pitch, and yaw angles (❸). Thus, RV software targets incorrect physical states (❹). In turn, the RVs lose attitude control due to the incorrectly set throttles of actuators (❺).

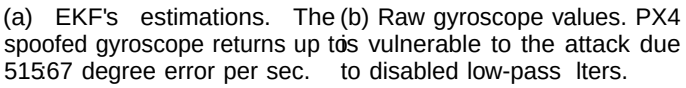
We identify seven different types of components within the attack-step model as prerequisites: RV types, sensor configurations, RV control software, software versions, flight modes, configuration parameters, and environmental conditions.

We note that these prerequisites partially overlap due to their hierarchical interdependence (e.g., RV control software implies a possible set of software versions). Yet, we separate prerequisites based on whether they can independently make attacks succeed or fail. For instance, each RV control software package configures hardware (e.g., sensors) in a peculiar way, regardless of its software versions. In parallel, the software version influences the attack success. Thus, RV control software and its software version independently changes the attack's success.

❶ **RV Types.** Attacks may require a specific RV type for two main reasons. First, signals at a high frequency (e.g., GPS L1 band is 1575.42 MHz) can only be effectively propagated through a specific type of medium, like air. Yet, such signals are unlikely to penetrate other types of mediums, like water [84]. Thus, some attacks, such as injecting fake GPS signals, cannot affect certain RV types (e.g., unmanned underwater robots [16]). Second, each RV type leverages a different coordinate system when the RV measures its attitude. This difference makes attacks fail to cause negative impacts. In particular, drones in the air and robots underwater measure their attitudes in a 3-D coordinate system. On the contrary, rovers on the ground are operated in a 2-D coordinate system, i.e., they do not use roll and pitch angles. Thus, rovers heavily depend on position sensors (e.g., GPS/GNSS receiver and optical flow sensor) but do not mainly rely on IMUs. Such a difference in the coordinate system makes the rover immune to disturbances in the IMUs. Indeed, as shown in Figure 4a, the drone crashes on the ground under an acoustic noise injection attack (the purple dashed line in Figure 4a). Yet, the rover does not face negative impacts (the cyan straight line in Figure 4a) under the same attack.

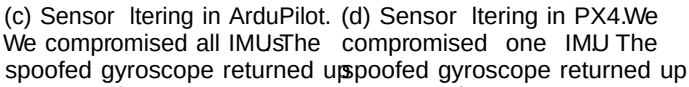
❷ **Sensor Configurations.** Attacks may require an RV that is equipped with a target sensor but is not equipped

2. Attackers can first jam legitimate GNSS signals and then spoof fake GPS signals. Yet, we note that we conduct GPS spoofing without the help of jamming according to steps reported in [97]



(a) EKF's estimations. The spoofed gyroscope returns up to 515.67 degree error per sec.

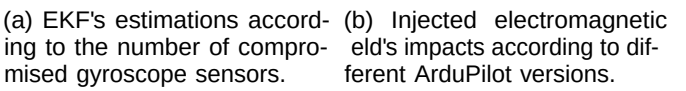
(b) Raw gyroscope values. PX4 is vulnerable to the attack due to disabled low-pass filters.



(c) Sensor filtering in ArduPilot. We compromised all IMUs. The spoofed gyroscope returned up to 538.15 degree error per sec.

(d) Sensor filtering in PX4. We compromised one IMU. The spoofed gyroscope returned up to 601.1 degree error per sec.

Figure 4: Impacts of acoustic noises in ArduPilot v.4.3.5 and PX4 v.1.13.0 The red area denotes the time under attack. Only the PX4 drone physically crashes on the ground.



(a) EKF's estimations according to the number of compromised gyroscope sensors.

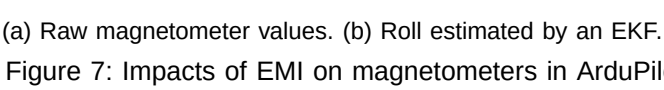
(b) Injected electromagnetic field's impacts according to different ArduPilot versions.

Figure 5: Impacts of (a) acoustic noise injection attacks and (b) EMI on magnetometers.

with a secondary sensor. This is because the RV can still correctly measure sensor values from the secondary sensor under attack. Indeed, compromising one or two of the three gyroscopes in an RV cannot cause significant physical impacts (black and yellow straight lines in Figure 5a). The RV shows 20 degrees of error at 18 seconds. Yet, the RV immediately recovers the stable roll angles because the RV's EKF switching logic (as explained in Section 2) stops using the compromised gyroscopes and starts using the intact ones.

To cause significant impacts, attackers need to spoof the primary and secondary sensors simultaneously. Yet, these multiple attacks require the adversaries to address signal interference issues [18], [87]. For instance, attackers may need to inject acoustic signals at multiple frequencies because (i) an RV is normally equipped with two/three different types of IMUs [65], and ii() each IMU sensor only can be compromised at a unique frequency [85], [88], [89], [93]. Yet, the acoustic signals at multiple frequencies can experience phase shifts and inversion of the signal polarity [18], [87].

Figure 6: Illustration of sensor filters in the most recent versions of each RV control software.



(a) Raw magnetometer values. (b) Roll estimated by an EKF.

Figure 7: Impacts of EMI on magnetometers in ArduPilot.

❸ RV Control Software. The effects of attacks completely vary depending on the RV control software for two main reasons. First, each RV software configures sensor parameters differently (e.g., sensor filters). Thus, the sensor measurements are also different, even under the same attack. PX4 [66] disables low-pass filters inside IMUs (Listing 1 in Appendix). On the contrary, ArduPilot and Paparazzi leverage low-pass filters inside IMUs (Listing 2 and Listing 3 in Appendix). Thus, as shown in Figure 4b, impacts of acoustic injection attacks are different according to RV software, although we conduct the same attack. Second, each RV software implements its sensor filtering algorithms differently, as shown in Figure 6. Thus, the impacts of attacks are also different. Compared to the complementary filters in MultiWii v.2.4 [58], EKFs show better filtering performance at high levels of disturbance [27], [61]. Thus, MultiWii is more vulnerable to attacks than others. Interestingly, ArduPilot, PX4, and Paparazzi leverage the same types of filtering algorithms (i.e., harmonic notch, low-pass, and range filters). Yet, the developers of each RV software implement the filtering algorithms differently.

For instance, dynamic harmonic notch filters [11], [76] automatically determine the primary noise frequency and adjust the notch's center frequency to filter out the noise. The harmonic notch filters in PX4 and Paparazzi can find and attenuate the maximum nine peak frequencies [76]. On the contrary, ArduPilot's harmonic notch filters track and attenuate the maximum 12 peak frequencies [11]. Thus, ArduPilot more effectively filters out spoofed sensor values compared to other RV control software. Indeed, as shown in Figure 4d, PX4's filters fail to filter out compromised gyroscope values, although we compromise only one IMU. Thus, the RV crashes on the ground at around 9 seconds (green line in Figure 4d). On the contrary, ArduPilot's filters stably filter out compromised gyroscope values, although we compromise all IMUs (green line in Figure 4c).

Figure 8: Sensor filters in different versions of ArduPilot.

❹ Software Versions. The attacks require specific software versions for two main reasons. First, RV developers have improved sensor filters while updating software versions. Second, they have implemented fail-safe logic [7], [62], [71] in recent software versions.

As of this writing, the latest versions of RV control software (e.g., ArduPilot v.4.3.5) are equipped with multiple levels of sensor filtering and improved sensor fusion algorithms, as shown in Figure 6. Such facts make it hard for an RV to retain compromised sensor values. For instance, attackers can inject electromagnetic fields to disturb an RV's magnetometers [82], as shown in Figure 7a. Yet, ArduPilot's range filter and EKFs filter out the spoofed sensor values, and the attack causes only around two degrees per second errors in roll angles (red line in Figure 7b). On the contrary, as shown in Figure 8, old versions of RV software (ArduPilot v.3.6.0 and v.3.2.1) leverage a single level of sensor filtering or a less accurate sensor fusion algorithm (e.g., direction cosine matrix). Such facts allow attackers to easily spoof/jam sensors. Indeed, as shown in Figure 5b, ArduPilot v.4.3.5 does not show any negative symptoms under the attack [82]. On the contrary, ArduPilot v.3.6.0 incorrectly decreases its altitude, although the RV must maintain the current altitude. The RV, which uses ArduPilot v.3.2.1, crashes on the ground due to its poor filtering algorithms [14] (Figure 8).

Moreover, RV control software is equipped with fail-safe logic [7], [62], [71] designed to recover RVs from unexpected sensor disturbances. For instance, GPS glitch fail-safe logic [34] in ArduPilot discontinues using the GNSS data and switches to rely on positions predicted by IMUs when the disparity between positions estimated by GNSS and IMUs surpasses the threshold. Thus, the fail-safe logic can mitigate attacks. Yet, old versions of RV software, unfortunately, do not have such fail-safe logic (e.g., ArduPilot v.3.2.1). In particular, the fail-safe logic can (i) stop using the primary sensor and start using the secondary one, or (ii) change the RV's flight mode into a manual mode, requiring the user to manually operate the RV. Thus, the RV continues its mission through the secondary sensor or the user's manual inputs.

❺ Flight Modes. Attacks require specific flight modes because an RV leverages different sensors or even stops using sensors according to its current flight mode. For instance, an RV automatically operates during Loiter mode [50], which uses all types of sensors attached to the RV. Conversely, Acro mode [2] operates based on a user's remote controller inputs rather than sensor readings. Thus, as shown by the

(a) Roll estimated by an EKF. (b) Pitch estimated by an EKF.

Figure 9: Impacts of EMI attack on SPI/I2C buses [42] according to flight modes in ArduPilot v.4.3.5.

blue lines in Figure 9, attacks (e.g., EMI injection to corrupt all different types of sensor measurements [42]) are hard to cause negative impacts when the RV is in Acro mode.

❻ Configuration Parameters. The attacks require specific configurations related to the sensor filtering and sensor fusion parameters for two main reasons. First, the configuration parameters determine the cutoff frequencies in the sensor filters in Figure 6 and Figure 8. The spoofed sensor values can be filtered out depending on the cutoff frequencies [11], [76]. Second, the configuration parameters determine each sensor's weight for the sensor fusion algorithms (i.e., EKFs). It is hard for an attack to cause negative impacts if attackers spoof/jam a sensor that an RV uses with a low weight. For example, as shown in Figure 7b, the injected electromagnetic field [82] causes around two degrees per second errors in roll angles (red line in Figure 7b). Yet, when we change ArduPilot's MAG_M_NSE configuration parameter [51] to make the RV reduce the weight of magnetometers and increase the weight of the gyroscopes, the attack fails to cause any negative impact (green line in Figure 7b).

❼ Environmental Conditions. Attacks need specific environmental prerequisites. This is because each sensor requires a unique environmental condition to correctly measure its values; thus, the impacts of the attacks also change according to the environment. For example, both optical flow sensors and GNSS receivers are used to measure an RV's current position. Yet, optical flow sensors fail to calculate changes in positions on floors without feature points (e.g., deserts) [22]. Similarly, GNSS receivers fail to calculate precise positions in an urban canyon due to multipath issues [39].

Attackers can exploit such a fact to achieve their goals. To illustrate, as demonstrated in Section 4, GPS spoofing fails to deceive the GNSS receiver because the GNSS receiver uses data from multiple satellite constellations. Yet, in specific scenarios where a target RV is in proximity to man-made structures, GPS spoofing can successfully achieve its attack goals. This is because the RV's GNSS receiver is hard-pressed to receive data from benign satellites, but it receives data from fake satellite signals created by the attackers.

6. Design Challenges

Our goal is to infer the prerequisites that make attacks possible; however, this process raises three unique challenges.

C1: Simulating Attacks. The first challenge is simulating attacks accurately. The simulators create sensor values with the following steps: (1) randomly select a sensor value, (2) (

Figure 10: Overview of RVPROBER's workflow.

add noises created by a standard normal distribution to the sensor value, and (3) feed the sensor value into filters. RV developers simulate sensor disturbances by adjusting the noise levels in the second step. Yet, attacks in the real world can have varying effects and may not follow a standard normal distribution due to different hardware and attack strategies used by attackers. To address this, RVPROBER first takes, as input, sensor values compromised under the attack; it then mutates the compromised sensor values based on four attack parameters: intensity, duration, start time, and attack algorithm.

C2: Natural Deviations. The second challenge concerns how to distinguish attack failure and success from an RV's physical states (e.g., position errors). This task is challenging because the RV's operating conditions may naturally cause negative effects regardless of the attacks. For instance, the GNSS receiver may show more than 15 meters of position errors without any attacks when there is a multipath issue [39] in an urban canyon. Further, the RV can deviate from a planned navigation path or even crash on the ground due to naturally occurring sensor glitches and wind gusts. To address this issue, RVPROBER first runs the same test case with and without an attack. It then extracts the RV's altered physical states caused by the attacks.

C3: High-dimensional Search Spaces. Each prerequisite creates an expansive search space. Thus, the last challenge is the high-dimensional search spaces of seven prerequisites and four attack parameters. In particular, the following three prerequisites create high-dimensional search spaces: software versions, configuration parameters, and environmental conditions (e.g., ArduPilot has 15,840 software versions, 1,140 parameters, and 168 environmental conditions). The combinations of these prerequisites are almost boundless. Thus, a method is required to reduce the search space while discovering the combinations of the prerequisites and attack parameters that make the attack possible. To overcome this challenge, RVPROBER leverages a binary search algorithm to probe software versions. Additionally, RVPROBER extracts relevant configuration parameters and environmental conditions through static and dynamic analysis; it then tests only the extracted ones.

7. RVPROBER

We introduce RVPROBER, a framework for probing the seven types of prerequisites required for attacks. Figure 10 shows the four main parts of RVPROBER. First, it takes, as input, a type of attack, its corresponding goal, and sensor values compromised by the attack. Second, it finds an appropriate set of attack parameters to achieve the goal. Third, it narrows down the search space by excluding particular configuration parameters and environmental conditions (from attack prerequisites) that are not relevant to the attack. Fourth, it mutates seven different types of prerequisites to discover the prerequisites required for conducting attacks successfully.

7.1. Inputs

RVPROBER takes, as input, a type of attack, its goal, and the sensor values compromised by the attack (1a and 1b in Figure 10). The users can select one of the 12 attacks in Table 3 and one of the three attack goals (mission failure, unstable attitude/position, or physical crash) in Table 2. Here, the users can optionally input a threshold for unstable attitude/position. For example, $\{\text{position}: 10\ \text{meters}\}$ denotes that RVPROBER will consider more than 10 meters of deviated positions as successful "unstable position" attacks.

We use the compromised sensor values as a guide to find the sensor values required to achieve attack goals in our setting. Users can obtain compromised sensor values taken during an attack with two different methods. First, the users can directly obtain compromised sensor values captured during the attack if they have the hardware and software required to conduct the attack. Second, they can obtain a log file taken during the attack. Then, they upload the file into log analysis software [8], [73] and obtain compromised sensor values stored in the log file.

7.2. Attack Profiling

This step's goal is twofold: (i) identifying physical states (e.g., position) affected by the attack (selected in Section 7.1) and (ii) determining the appropriate attack parameters necessary to achieve the attack goal on simulators.

Monitoring Physical States. The 12 attacks aim to cause physical crashes, instability, and mission failure. Thus, to