ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20241210002187
Original scientific paper

Enhancing UAV Security through Comprehensive Vulnerability Analysis and an ARM PSA-Based Architecture

Beom Seok KIM, Heeseung SON, Junoh LEE, Daehee JANG, Deokjin KIM, Sangwook LEE, Ben LEE, Jinsung CHO*

Abstract: Recently, Unmanned Aerial Vehicles (UAVs), which are embedded, real-time systems, are being used in various fields such as military, logistics, surveillance, and mapping. UAVs face security threats due to their diverse peripherals and attack vectors. Real-world incidents, such as GPS spoofing and control signal manipulation, highlight the need for improved security systems in UAVs to prevent potential human losses and material damage. While prior research has examined UAV security vulnerabilities, it has primarily focused on individual attack techniques rather than systematically analyzing the relationship between software and hardware security risks. As such, effectively addressing these security threats necessitates a comprehensive analysis of the software and hardware architecture of UAVs and a systematic investigation of existing attack cases. This paper systematically analyzes UAV attack cases and defines a generalized UAV architecture based on commercial and open-source UAV platforms. Furthermore, we evaluate the security vulnerabilities in widely used UAV software frameworks, including PX4 Autopilot and the MAVLink communication protocol. Based on our findings, we propose a Secure UAV architecture leveraging ARM Platform Security Architecture (PSA) to enhance resilience against cyber threats.

Keywords: ARM PSA; security analysis; secure platform model; UAV

1 INTRODUCTION

UAVs are a specialized category of Internet of Things (IoT) devices that perform autonomous and remote-controlled missions [1]. Unlike typical IoT devices, UAVs operate in dynamic environments and integrate diverse peripherals, including GPS modules, inertial measurement units (IMUs), cameras, and wireless communication systems. This extensive integration introduces a significantly larger attack surface, making UAVs more vulnerable to cyber threats than conventional IoT devices. The reliance of UAVs on wireless communication and external control systems further exposes them to security risks such as GPS spoofing, jamming, and control hijacking. Real-world incidents, such as the 2011 Iranian military UAV hijacking [2] and the 2014 WiFi deauthentication attack on a UAV during an Australian triathlon event [3], highlight the critical need for robust security mechanisms.

As UAVs become more complex, their security vulnerabilities expand due to the various communication protocols and hardware configurations they incorporate. Their ability to operate autonomously in critical missions makes their security breaches particularly severe, as they can lead to unauthorized access, mission disruption, and even physical damage. These risks emphasize the need for comprehensive UAV security measures that address hardware and software vulnerabilities.

Previous research has explored UAV security vulnerabilities; however, most studies have focused on isolated attack techniques rather than analyzing security risks in relation to the overall UAV system architecture. While individual attack types, such as Denial of Service (DoS), spoofing, tampering, and data leakage, have been examined [4-23], these analyses often lack a systematic evaluation of how hardware and software vulnerabilities interact within UAV ecosystems. Therefore, a more holistic approach is needed to address UAV security effectively.

To mitigate these risks, it is crucial to understand the software and hardware architecture of UAVs as a foundational step in identifying and classifying

UAV-specific security threats. Building on this analysis, a structured investigation of existing attack cases must be conducted, followed by developing an integrated security framework that encompasses vulnerability identification and mitigation strategies. Additionally, the variety of UAV designs and manufacturers complicates the implementation of uniform security standards, making it essential to derive a generalized UAV security architecture.

The key challenges in UAV security include:

Heterogeneous Architectures: UAVs from different manufacturers have diverse hardware and software configurations, making generalized security evaluations difficult.

Varying Security Needs: Security services for UAVs depend on their specific hardware and intended use cases.

Integration of Security Measures: A reliable security framework must be derived based on UAV hardware and software requirements, followed by a structured attack case analysis.

Building upon the challenges identified above, this paper makes the following key contributions:

Generalized UAV Architecture Definition:

A generalized UAV architecture is derived, incorporating both commercial UAVs and open-source platforms such as PX4 Autopilot and MAVLink, forming the foundation for systematic security evaluations.

Vulnerability Assessment of UAV Software Platforms:

A security analysis of widely used UAV software frameworks, particularly PX4 Autopilot and MAVLink, is performed to identify critical vulnerabilities at the code level.

Comprehensive UAV Security Analysis:

A systematic analysis of real-world UAV attack cases is conducted, categorizing security threats based on their impact on UAV components, communication systems, and firmware.

Proposed Secure UAV Architecture:

Leveraging prior research on the security platform model [24] designed for IoT devices, the model is customized to meet the unique security requirements of UAVs. The proposed secure UAV architecture is based on ARM Platform Security Architecture (PSA) and integrates trusted execution environments, cryptographic protection mechanisms, and real-time security monitoring.

Framework for Future UAV Security Implementations:

A structured foundation for future UAV security research is provided, enabling researchers and manufacturers to implement standardized security measures effectively.

The proposed secure UAV architecture builds upon a previously developed security platform model, adapting it to address the security challenges inherent in UAV systems. By integrating ARM PSA-based security mechanisms and optimizing the model for UAV-specific vulnerabilities, this framework enhances UAV resilience against cyber threats. This architecture ensures not only secure execution environments but also scalable and adaptable security measures that can evolve with emerging UAV security demands. This research establishes a comprehensive foundation for advancing UAV security, ensuring safer and more robust UAV operations in diverse applications.

The organization of the paper is as follows. Section 2 defines a generalized UAV architecture by analyzing commercial UAV designs and widely used open-source platforms, such as PX4 Autopilot and MAVLink, to establish a structured foundation for UAV security evaluations. Section 3 describes a systematic analysis of attack cases based on the derived UAV system and the general UAV architecture. Section 4 evaluates the vulnerabilities identified by analyzing attack cases targeting PX4 Autopilot and MAVLink. Section 5 proposes a secure UAV architecture based on ARM PSA and PX4 Autopilot as an effective solution to address the identified vulnerabilities. Finally, Section 6 concludes the paper and discusses possible future work.

2 UAV SYSTEM AND HARDWARE/SOFTWARE ARCHITECTURE

2.1 The UAV System

Fig. 1 illustrates the UAV system. The most basic system consists of a UAV and a Remote Controller (RC). In addition to basic flight capabilities, UAVs have additional features, such as cameras, robotic arms, and various onboard sensors, including LiDAR, infrared cameras, and environmental monitoring modules, enabling a wider range of applications.

UAVs can also be controlled using Ground Control Systems (GCS), which serve as the primary interface for mission planning, flight monitoring, and data analysis. These systems can be operated through smartphones and laptops for basic control, while dedicated GCS devices provide enhanced processing power and security features for professional and military applications.

The most complex form of a UAV system is designed for specific purposes, such as military and logistics operations. These systems typically consist of multiple UAVs coordinating with a GCS and a central server. The central server is crucial in mission planning, real-time data processing, and inter-UAV communication, ensuring efficient coordination and decision-making in large-scale UAV deployments. Each component interacts with others through communication technologies such as RC and WiFi.

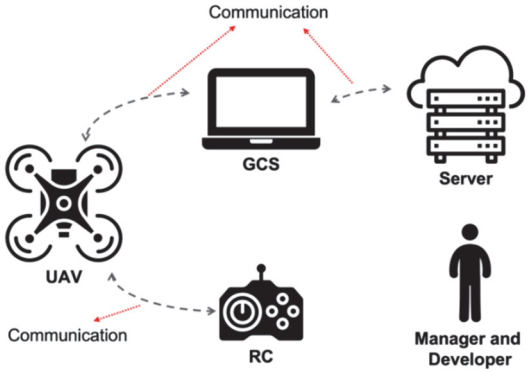

Figure 1 The UAV system

Furthermore, UAV developers, administrators, and dedicated update servers are vital in UAV development and maintenance. Firmware and application updates are essential for enhancing UAV functionality, patching security vulnerabilities, and ensuring compliance with evolving operational requirements. Secure update mechanisms are critical to preventing unauthorized modifications and potential cyber threats.

2.2 The Basic and Extended UAV Hardware Model

The UAV is the most important component of a UAV system. To perform a comprehensive security analysis, existing commercial UAVs with diverse hardware architectures were examined, and a generalized UAV hardware architecture was derived. Three widely used commercial UAVs were selected to validate the derived general UAV hardware architecture, and their structures were analyzed through disassembly.

In general, UAV hardware architecture can be classified into either the basic UAV hardware model consisting of essential hardware components, or the extended UAV hardware model that includes optional hardware as shown in Fig. 2. The basic UAV hardware model shown on the left side of Fig. 2 comprises the Flight Control Board (FCB) to control the peripherals necessary for flight operations. The key components include actuators composed of motors and Electric Speed Control (ESC), sensors such as GPS, Inertial Measurement Unit (IMU), barometer, anemometer, and optical flow for the flight path and attitude control, as well as communication modules like Radio Frequency (RF), telemetry radio, WiFi, and cellular for remote control capabilities. The extended UAV hardware model shown on the right side of Fig. 2 is an expanded form that adds a Mission Computer, which is responsible for managing specialized functionalities required for mission execution in domains such as military and logistics. These additional computational resources enhance the UAV's ability to process sensor data, perform real-time decision-making, and execute advanced tasks autonomously.

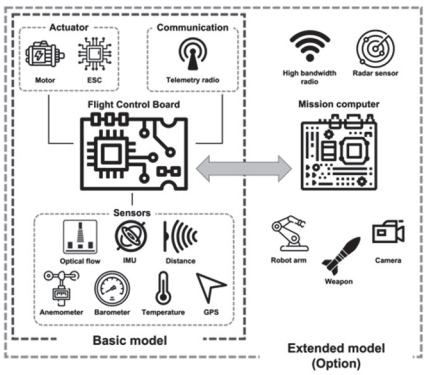

Figure 2 A basic and extended UAV hardware model

To validate the basic and extended UAV hardware models, three widely used commercial UAVs were selected for disassembly and analysis. The details of the selected UAVs are as follows:

Table 1 Boards and functionalities of DJI Phantom 4 Pro V2.0

Board	Function
FCB	Flight control through sensor value-based ESC (Electronic Speed Control) commands
Main processor board	Mission computer to control and manage all optional functions
Camera encoder board	Receive and compress camera data, prepare FPV video stream
Gimbal master control board	Gimbal main control and inertia measurement
Gimbal power and SD PTZ board	Power adjustment for camera, compressed video stream and photo recording
Gimbal rool yaw motor ESC board	ESC control for gimbal motor
GPS module board	Receive GPS signal
VPS sonar board	Altitude measurement by ultrasound
Aircraft power port board	Supply power to other boards from battery, transmit communication between boards
Proximity sensor board	Object proximity measurement via infrared
Antenna and compass leg board	Provides antenna and direction data
Battery module board	Battery management

DJI Phantom 4 Pro V2.0: This is a representative high-performance UAV that offers a wide range of features [25] and aligns with the extended UAV hardware model. Tab. 1 includes descriptions of the boards that constitute the DJI Phantom 4 Pro V2.0. It utilizes the FCB and various sensors to perform essential flight functions corresponding to the basic UAV hardware model. In addition to the flight-related functionalities, DJI Phantom 4 Pro V2.0 includes dedicated boards to support additional functions such as video recording, gimbal control, and transmission of video and flight data to the GCS. The Main Processor Board, functioning as the Mission Computer, is powered by an ARM Cortex-A7 MCU and operates on an Android 4.4-based system.

NXP Hovergames: This is a UAV in the form of a development kit and runs an open-source firmware installed on the Flight Management Unit (FMU), which represents the FCB [26]. Hovergames utilizes the RDDRONE-FMUK66 development board equipped with an ARM Cortex M4Fbased MCU as the FMU. The FMU also includes RF communication modules, motors, batteries, and GPS sensors. The FMU as the core component implements the essential hardware elements required for UAV flight, such as sensors, motor control, communication, and sensor data calibration. It is also designed to accommodate additional modules, such as cameras.

Syma X5C: This is the simplest form of a UAV among commercial UAVs [27]. The FCB is centered around an ARM Cortex M0 MCU and is connected to RF communication modules, sensors for flight, motors, and batteries. In addition, it has a camera and an SD card slot for storing captured photos and video data.

The comparative analysis of these UAVs provides valuable insights into the commonalities and variations in UAV hardware architectures. The disassembled and analyzed structures of the three representative commercial UAVs were found to exhibit similarities with the basic and extended UAV hardware models. This analysis confirms the validity of the generalized UAV hardware architecture and its applicability across different UAV platforms.

2.3 The Open-Source Platform for UAV: PX4 Autopilot and MAVLink

As previously discussed, UAVs incorporate diverse hardware configurations, leading to variations in their software architectures. Therefore, it is necessary to derive a general software structure to design effective countermeasures against vulnerabilities. This paper analyzes open source platforms for UAVs, such as the PX4 Autopilot software platform [28] and the MAVLink communication protocol [29] for intra- and inter-UAV communications, to provide a foundation for the vulnerability analysis.

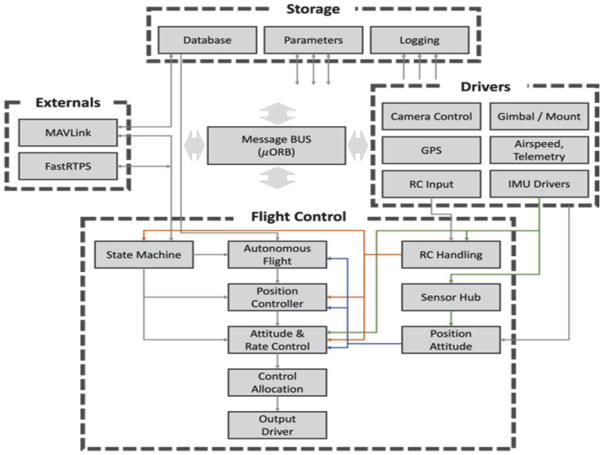

Figure 3 Software architecture of PX4 autopilot

PX4 Autopilot: Fig. 3 shows the PX4 Autopilot or Pixhawk project, which is an open-source autopilot system for remotely controlled drones and UAVs. It was originally developed as part of the Ardupilot ecosystem but has since evolved into an independent project. The firmware embedded in the FCB consists of an Externals module for communication with external entities, a Drivers module for hardware control, a Storage module for storing flight information and logs, and a Flight Control module for flight control operations. Each module operates based on the NuttX real-time operating system and interacts through the Message BUS, which is a publish/subscribe-based middleware called μORB. The Flight Control module receives flight-related data collected through the Sensor Hub, which includes sensor measurements of the UAV's

flight environment. The Flight Control module analyzes the UAV's current flight state information and performs attitude control. The flight data is forwarded to the Autonomous Flight, Position Controller, and Attitude & Rate Controller. Afterwards, the Control Allocation verifies whether the processed flight data obtained in the preceding stages can be controlled and subsequently transmits the corresponding values to the Output Driver. Based on this information, the Output Driver directly controls the motors and ESCs. This modular approach enhances real-time performance and ensures flexibility when integrating different UAV hardware configurations.

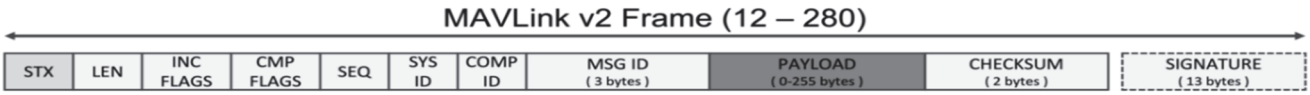

Figure 4 Message structure of MAVLink

Drivers for various sensors, including GPS and communication modules, such as MAVLink and FastRTPS for remote control and supervision, acquire measurements of data required for flight. In addition, the Storage module facilitates the storage of essential flight parameters and operational logs for performance tracking and post-flight analysis. The Message BUS connects all these components allowing interactions among them for the overall operation of the UAV.

MAVLink: MAVLink is a lightweight messaging protocol that supports communication between UAVs and external components like GCS. It is designed to be compatible with multiple communication protocols, including TCP and UDP, and supports interfaces such as WiFi, telemetry radios, and mobile networks. Fig. 4 illustrates the message structure of MAVLink v2, which contains essential information required for communication between UAVs and external components. This structured message format ensures reliable command transmission and data exchange between UAVs and their control systems. In particular, it offers the option to include a signature for the message to ensure the integrity of transmitted messages. This cryptographic feature helps prevent unauthorized modifications and replay attacks, thereby enhancing the security of UAV communications.

2.4 The General Architecture of UAVs

Based on the commercial and open-source UAV architectures discussed earlier, the general UAV architecture can be defined as shown in Fig. 5, which consists of three main components: hardware, software, and peripherals. This architecture provides a structured foundation for understanding UAV system design and security considerations.A detailed description of each component is as follows:

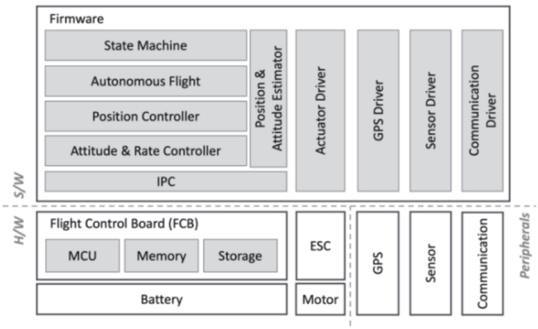

Figure 5 The general UAV architecture

Hardware: The hardware represents the essential components required to operate the UAV. This includes the FCB to perform the necessary computations for flight control, the ESC for controlling high-power motors, and the battery. In addition, advanced UAVs may incorporate redundant power systems, custom processing units, and specialized sensors to enhance reliability and adaptability in various applications.

Software: The software is typically implemented as a firmware that supports essential functionalities for UAV control. The firmware consists of various modules including the State Machine, Autonomous Flight, Position Controller, Attitude & Rate Controller, Position & Attitude Estimator, Inter-Process Communication (IPC), and drivers for peripherals. The State Machine module oversees and controls the mission to be performed by the UAV, while the Autonomous Flight Controller controls the mission navigation defined by the State Machine. The Position Controller performs mission navigation controlled by the Autonomous Controller for precise mission execution. The Position & Altitude Estimator receives and calibrates raw sensor values from the peripherals to measure the current attitude and position of the UAV. The Attitude & Rate Controller receives the desired attitude values from the Position Controller and the current attitude information from the Position & Attitude Estimator. Based on this information, the ESC controls the motors to manipulate the UAV. To ensure real-time responsiveness and fault tolerance, the software architecture is designed to support modular updates and dynamic reconfiguration. The drivers corresponding to each of the peripherals receive measurement values required for UAV operation and transmit them to the respective UAV control modules via IPC.

Peripherals: Peripherals include the communication module for remotely receiving user commands, sensor modules such as anemometers, barometers, and Inertial Measurement Units (IMUs) for measuring the UAV's current position and attitude, and the GPS module for receiving position information from satellites. In addition to basic peripherals, mission-specific UAVs may integrate advanced payload systems, including thermal imaging cameras, LiDAR sensors, and radar modules, for enhanced situational awareness and operational flexibility.

The general UAV architecture in Fig. 5 shows that UAVs are equipped with a multitude of peripherals, unlike common IoT devices. These diverse peripherals are managed through a structured software framework, enabling seamless integration and coordinated operation during flight. Furthermore, the general UAV architecture

contains important elements from both existing commercial UAVs and open-source UAV platforms. As a result, it has the potential to serve as a reference model not only for future UAV hardware and software development but also for enhancing security measures. By leveraging this architecture as a standard framework, UAV developers can systematically address cybersecurity challenges, improve interoperability, and facilitate the development of more resilient UAV systems.

3 A COMPREHENSIVE ANALYSIS OF ATTACK CASES AGAINST UAVS

This section analyzes attack cases targeting existing UAVs based on the general UAV architecture defined in Subsection 2.4

3.1 Case Studies

With the increasing occurrence of attacks targeting operational UAVs, the need for enhancing their security has become critical. Recent security incidents demonstrate that UAV vulnerabilities can be exploited for unauthorized access, mission disruption, and data interception, highlighting the necessity for comprehensive security evaluations. There have been several recent research efforts that focused on analyzing such attack cases as outlined below:

Walters published 30 drone vulnerability cases on the online blogging platform medium [4]. This page was updated with new drone vulnerability details from diverse sources until August 2019 including internet blogs, news articles, DEFCON, YouTube, and research papers.

The RAND Corporation, which is a U.S. policy research institute, published a report that explored the understanding of UAV cybersecurity, record surveys, and approaches for modeling [5]. The report provides a table categorizing 47 drone attacks using the STRIDE technique, which is a security threat modeling scheme developed by Microsoft [30].

Chaari et al. investigated the vulnerabilities in

UAV-to-GCS communication and specifically identified and analyzed vulnerabilities in the most widely used UAV communication protocol, MAVLink [6]. Their research classified UAV security vulnerabilities into 20 distinct categories, covering aspects such as authenticity, confidentiality, integrity, and availability.

Yaacoub et al. investigated the threats posed by increasing cyberattacks using drones and corresponding countermeasures [7]. They categorized the drone attacks into 22 types based on attack characteristics, degree of intrusion, and encryption/non-encryption countermeasures.

In addition to these studies, several recent research efforts have focused on UAV vulnerabilities, particularly emphasizing real-world attack vectors and software/hardware security flaws. This paper includes an analysis of the recent studies that focus on UAV vulnerabilities with an emphasis on the latest reports related to UAVs from the Common Vulnerability Exposure (CVE), which are publicly disclosed vulnerabilities [8-23]. In particular, vulnerabilities related to the MAVLink protocol are prioritized. Among these, vulnerabilities associated with the MAVLink protocol are given particular attention due to their widespread usage and potential security implications.

Table 2 The recent studies that focus on UAV vulnerabilities

Reference	Abstract
[8]	DTLS (Data Transport Layer Security) protocol to capsulate MAVLink protocol
[9]	Network IDS through MAVLink MSG ID Sequence and LSTM (Long-Short Term Memory) RRN (Recurrent Neural Network)
[10]	FPGA module based on ARM Trust Zone and PUF (Physically Unclonable Function)
[11]	Hardware-based security module for secure communication
[12]	MAVSec to protect MAVLink message between UAV and GCS
[13]	Policy-Guided Fuzzing for Open-source UAV firmware
[14]	Vulnerability attack tools for commercial UAVs
[15]	Experiment to disable UAVs with intentional noise sound for MEMS gyroscopes
[16]	Crowd-GPS-sec to remotely detect and specify GPS spoofing and the location of the spoofing device
[17]	RVFuzzer looking for validation bugs with control-guided input mutation in RV communication
[18]	Report of buffer overflow and WiFi deauthentication packet vulnerabilities in Parrot ANAFI (Commercial UAV)
[19]	Attack tools to DJI Tello and DJI Mavic 2 drones (Replay, GPS spoofing, Side channel attack)
[20]	Performing GPS Spoofing attack using SDR targeting DJI Phantom 3
[21]	Flight data extraction for DJI Mavic Air 2
[22]	Telnet connection and packet sniffing for Parrot AR Drone 2.0, Parrot Disco FPV, DJI Spark, and DJI Robomaster
[23]	Analyzing Parrot AR Drone 2.0 software vulnerabilities and suggesting solutions

While these studies provide valuable insights into UAV security risks, they primarily focus on attack case collection and classification, without performing a deep analysis of the underlying software and hardware architectures of UAVs.

3.2 Detailed Analysis

This subsection provides a more detailed analysis of the previously examined attack cases targeting UAVs. Out of 193 total cases analyzed, 190 cases involve attacks exploiting vulnerabilities in UAVs while the remaining 3 cases focus on UAVs being leveraged as attack vectors against external systems. The latter cases fall outside the scope of this paper as they do not address vulnerabilities within UAVs. Duplicate cases from previous studies were excluded, and in cases where multiple targets were involved in a single case, each target was classified separately.

The analysis results are presented in Tab. 3, which are categorized based on the hardware and software components of the general UAV architecture defined in Subsection 2.4. The classification considers the attack targets, components, types, outcomes, and STRIDE. Each criterion is described in detail below:

Target: Attacks are classified based on the following targets: internal components of UAVs (UAV), RC, smartphones/laptops (GCS), vulnerabilities in servers of UAV manufacturers (Server), and non-technical attacks exploiting human interaction and trust to gather information (Human).

Component: Refers to specific targets within a UAV and is classified into Firmware, Application, Sensor, GPS, Communication, and Human (uses and develops). Note that because GPS has a distinct role that differentiates it from other sensors and numerous attack cases exist, it is classified as an independent attack component.

Type: Represents attack types employed by the attacker and can be classified based on the targeted components. First, there are attacks that exploit vulnerabilities in communication technologies, such as Jamming, Deauthentication, Fabrication, Modification, Replay, Sniffing, and Flooding. Second, there are attacks that exploit software vulnerabilities that specifically target components like UAV firmware. These attacks include Firmware Modification, Reverse Engineering, Buffer Overflow, Password Attacks, and Malware. Lastly, there is an attack called Social Engineering, which capitalizes on the deep trust in human interactions to deceive developers or users to undermine normal security procedures. Tab. 4 provides explanations for the various attacks.

Result: Represents the attack outcomes and are classified as Malfunction (i.e., causing erroneous behavior), Information Disclosure (i.e., leaking UAV information), or Losing Control (i.e., rendering the UAV uncontrollable).

STRIDE: Attacks are classified based on the threat model proposed by Microsoft, which are Spoofing, Tampering, Repudiation, Information Disclosure, DoS, or Elevation of Privilege.

Examining the distribution of attack types by target and component, attacks targeting UAVs account for 170 cases representing approximately 89.47%. There are two cases on RC, 10 cases on Smartphones, 5 cases on Laptops, one case on Server, and two cases of misconduct by UAV system developers (i.e., Human).

Table3 Security analysis of existing attack cases for UAVs

Target		Component	Type	Result	STRIDE	# of cases	Ratio
UAV		Communication	Jamming	Malfunction	DoS	12	89.47
			Deauthentication	Losing control	Spoofing	4	
					Elevation of privilege	1	
				Malfunction	Spoofing	4	
					DoS	5	
			Fabrication	Losing control	Spoofing	22	
					DoS	1	
				Information disclosure	Spoofing	1	
					Tampering	1	
					Information disclosure	1	
				Malfunction	Spoofing	10	
					Tampering	1	
			Modification	Losing control	Tampering	2	
				Malfunction	Tampering	2	
			Replay	Malfunction	Spoofing	5	
			Sniffing	Information diclosure	Information disclosure	16	
			Flooding	Malfunction	DoS	9	
		GPS	Jamming	Malfunction	DoS	5	
			Fabrication	Losing control	Spoofing	5	
				Malfunction	Spoofing	10	
					DoS	1	
		Sensor	Jamming	Malfunction	DoS	2	
			Fabrication	Malfunction	Spoofing	2	
					DoS	1	
		Firmware	Modification	Losing control	Tampering	9	
					Information disclosure	2	
			Reverse engineering	Information diclosure	Information disclosure	4	
			Buffer overflow	Malfunction	Tampering	2	
				Losing control	Tampering	2	
			Password	Losing control	Elevation of privilege	17	
				Information diclosure	Information disclosure	3	
					Elevation of privilege	3	
			Malware	Losing control	Elevation of privilege	4	
				Malfunction	Tampering	1	
RC		Communication	Password	Losing control	Tampering	1	1.05
					Elevation of privilege	1	
GCS	Smartphone	Communication	Fabrication	Losing control	Spoofing	1	7.89
		GPS	Jamming	Malfunction	DoS	1	
			Fabrication	Losing control	Spoofing	1	
		Application	Modification	Losing control	Tampering	1	
			Reverse engineering	Information diclosure	Information disclosure	4	
			Malware	Losing control	Spoofing	1	
				Information diclosure	Spoofing	1	
	Laptop	Communication	Fabrication	Losing control	Spoofing	1	
		Application	Fabrication	Losing control	Spoofing	1	
			Malware	Information disclosure	Tampering	1	
					Information disclosure	1	
				Losing control	Elevation of privilege	1	
Server		Firmware	Reverse engineering	Information disclosure	Information disclosure	1	0.53
Human		Human	Social engineering	Information disclosure	Information disclosure	2	1.05

Most attacks directly target UAVs with 97 cases targeting communication methods such as RF, WiFi, and ad-hoc networks, followed by 47 cases targeting the firmware. This indicates that UAVs are most vulnerable to attacks on communication. Among these, the most straightforward yet difficult-to-avoid attacks are Jamming and Flooding with 12 and 9 cases, respectively. There are also 5 cases of Deauthentication attacks that simply render

the control ineffective. Excluding these, the remaining 71 cases involve attempts to compromise UAV control, induce malfunctions, or exploit security vulnerabilities in UAV systems. These attack cases can lead to direct attacks on firmware, and indeed, there are 47 cases of attacks targeting Firmware. In particular, any attacks targeting UAV firmware can potentially cause issues upon execution, and while these attacks may not be immediately noticeable during the attack moment, they can lead to significant security issues during UAV operations afterward.

Furthermore, there are 21 cases targeting GPS and 5 cases targeting sensors, which mostly involve attacks that generate erroneous GPS data and sensor measurements. The consequences of attacks resulting from abnormal sensor data lead to losing control and malfunctions. To address these issues, it is necessary to analyze the causes of abnormal sensor data and implement methods such as filtering and calibration to mitigate the impact of such malicious acts.

Table 4 Attack types for UAVs

Attack Type	Explanation
Jamming	Attacks that disrupt communication, GPS, or sensor functionality by transmitting wireless signals that reduce the signal-to-noise ratio.
Deauthentication	Attacks targeting WiFi, where deauthentication frames are sent to legitimate users causing them to disconnect from the network.
Fabrication	Attacks on communication, GPS, or sensor systems where an attacker masquerades entity A and transmits data to entity B.
Modification	Attacks that involve an attacker intercepting and modifying data transmitted from entity A to entity B in communication scenarios. In addition, it includes unauthorized modifications to firmware or applications resulting in malfunctions.
Replay	Attacks where an attacker intercepts and retransmits data transmitted from entity A to entity B.
Sniffing	Attacks where an attacker intercepts data transmitted from entity A to entity B.
Flooding	Attacks that overwhelm the target by transmitting data exceeding its processing capabilities within a given time frame.
Reverse Engineering	Attacks that involve extracting sensitive information, such as hardcoded security mechanisms or keys, from firmware or application code using reverse engineering techniques.
Buffer Overflow	Attacks that exploit vulnerabilities by inputting data that exceeds the allowed buffer size through external interfaces causing memory errors.
Password	Attacks that exploit easily predictable passwords through brute force or dictionary attacks \\to gain unauthorized access.
Malware	Attacks that involve inserting malicious programs into firmware or applications to enable unauthorized actions by the attacker.
Social Engineering	Attacks that exploit deep trust in human interaction to deceive developers or users and undermine \\established security procedures.

Smartphones and laptops serve as GCS for UAV operations, and GCS is a system designed to monitor UAV missions, thus requiring high security. Similar to communication attacks targeting UAVs, attacks on smartphones and laptops can directly compromise GCS. However, most GCS environments possess sufficient computing resources making them feasible to apply existing security services.

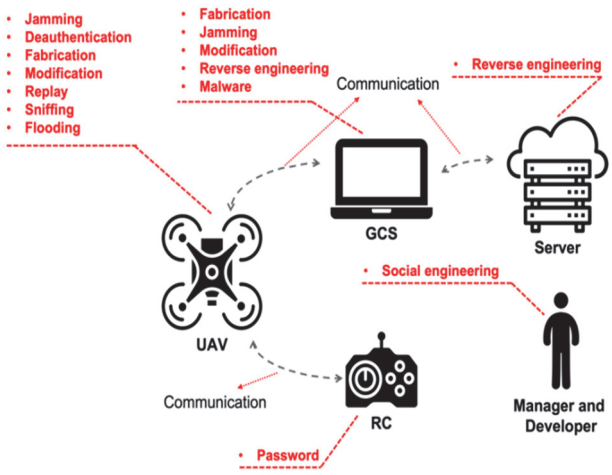

Figure 6 Vulnerabilities in the entire UAV systems

Attacks targeting servers that provide UAV firmware updates also exist. Therefore, robust security measures for servers are necessary to ensure secure firmware updates, including introducing security systems and applying secure services to guarantee the security of communication with UAVs.

Other attacks include those targeting the RC, where the attacker infers the password to seize control, and social engineering attacks resulting from developers' misconduct Fig. 7 maps identified vulnerabilities within the general UAV architecture, highlighting susceptibility in Communication (Replay, Jamming, Flooding), Firmware (Memory-based exploits, Reverse Engineering, Malware), and Navigation Systems (GPS and Sensor manipulation). The Actuator module was found to be relatively resistant to security threats.

4 A SECURITY ANALYSIS OF OPEN-SOURCE SOFTWARE FOR UAVS

In this section, the vulnerabilities identified in Section 3 are validated for their existence at the code level in the PX4 Autopilot. This is achieved by analyzing the PX4 Autopilot code to determine whether effective countermeasures are implemented to prevent each vulnerability.

Tab. 5 summarizes the vulnerabilities and implementation status of countermeasures for each module in the PX4 Autopilot. The table provides a structured overview of security gaps, highlighting areas where countermeasures are either fully implemented, partially implemented, or entirely absent. The implementation of countermeasures for the RC and MAVLink communication modules is separately indicated on the left and right sides, respectively. The symbol "O" indicates that countermeasures to address the vulnerability have been implemented, while "X" signifies that no countermeasures have been implemented. The symbol "△" denotes inadequate implementation indicating that the functionality to address the vulnerability has been implemented but may not be effective. In addition, the symbol "−" denotes the absence of RC vulnerabilities in the communication system

being evaluated. The following provides a detailed analysis:

GPS module: During the process of reading data from the GPS module, the inject Data () function is called from the handle Inject Data Topic () function, which in turn calls the dump Gps Data () function to transmit the GPS data. However, these functions lack mechanisms for discontinuous sensing data detection, signal strength monitoring, and cross-verification with other sensors, leaving them vulnerable to GPS spoofing and jamming. Furthermore, the GPS_Sat_Info structure representing the received GPS data conveys information about the satellite signals through the satellite_info_s structure in the μORB. However, there are no implementations of satellite signal number monitoring and satellite identification code-based error detection. In addition, to verify the validity of GPS data, the dump Gps Data() function records the timestamp when GPS data is dumped and the run() function reads the GPS data every 5 seconds. Although the run () function checks for time intervals greater than 5 seconds, this simple comparison cannot detect attacks if the timestamp is manipulated. Therefore, the time interval check in the run () function is not an effective countermeasure against attacks. This indicates that the GPS module is vulnerable to Fabrication and Jamming.

Communication module:

RC: In the code that controls the RC communication, the nonce, which is a random number used only once in a cryptographic communication, and the sequence number is not subject to any verification process. Instead, the received data is directly parsed and stored in the Rc_in structure. In the RC communication, the scan state representing the ability to receive messages is reset after receiving a message to await the next message. During this process, the timestamp is checked to validate the previously accepted message. In addition, Heartbeat is performed using the timestamp. However, if the connection with the RC is interrupted due to jamming or other attacks, the implemented functionality does not perform emergency take off or Return to Home (RtH) and instead continues to execute the previous command. In terms of transmission packet authentication, the data received from the SBUS is parsed and packet authentication should be performed before delivering the message to μORB. However, packet authentication is not implemented in the current PX4 Autopilot and the message is directly delivered to μORB after parsing. Furthermore, the algorithms related to packet encryption during the parsing process have not been implemented; therefore, even though error checking is performed, packet integrity checks are not performed on the data.

MAVLink: In the MAVLink *v*2 frame specification, nonce is not implemented, and while the sequence number field exists, there is no functionality to discard messages with invalid sequence numbers. The MAVLink *v*2 frame specification does includes a timestamp field, and similar to the RC, the code implements a mechanism to avoid receiving outdated messages based on the timestamp. The Heartbeat functionality also behaves similarly to the RC, where no specific handling is performed if the connection is lost and the previous command continues to execute. Unlike the RC, MAVLink *v*2 includes a message signature and performs packet authentication and integrity checks during the message signature analysis phase using the Hash-based Message Authentication Code (HMAC) algorithm. To apply encryption to transmitted/received messages in the MAVLink *v*2, an encryption/decryption process is required before message parsing. However, the current PX4 Autopilot does not implement packet encryption. In addition, MAVLink *v*2 does not implement credentials for SSH and FTP, which can be potential unauthorized access paths to the UAV. Moreover, IDS is necessary to prevent attacks such as flooding, but it should be implemented separately from the FCB due to performance issues. However, the current version of PX4 Autopilot does not implement IDS. In summary, the RC module does not implement all the countermeasures, except for timestamp and heartbeat. Similarly, MAVLink does not implement all the countermeasures except for sequence number, timestamp, packet authentication, and packet integrity checks.

Firmware: In general, all UAV operations rely on code implemented within its firmware, thus attacks targeting the firmware can lead to catastrophic consequences for the UAV mission execution. Therefore, ensuring firmware integrity and secure updates, as well as implementing security services to prevent malicious analysis and tampering of the firmware, are imperative for UAV security. For example, during the firmware update process, attackers can manipulate the firmware or insert malware, making it crucial to ensure the secure update of the firmware. However, the current PX4 Autopilot offers only simple updates through HTTP and serial connections without implementing a separate verification process. The *px4_fmu_v5* version of PX4 Autopilot can perform a secure boot by setting the SECURE_BLT_ENABLED flag, which includes firmware integrity checks. However, real−time remote attestation that allows for continuous integrity checks during UAV operation is not implemented. On the other hand, buffer overflow represents a well-known attack method targeting running firmware. To avoid this, security measures such as secure coding and Control Flow Integrity (CFI) are imperative. Secure coding involves validating internal function vulnerabilities such as out-of-bound and type confusion during firmware development. PX4 Autopilot utilizes external libraries like MAVLink and NuttX OS, which exposes vulnerabilities. The actual code also contains instances of out-of-bound vulnerabilities, which can be discovered and exploited by attackers using vulnerability testing methods like fuzzing. Another effective approach for safeguarding UAV firmware is CFI, which requires hardware and operating system support. However, PX4 Autopilot that employs external libraries and third-party OS does not provide the CFI support. Meanwhile, for firmware stored in flash memory, obfuscation techniques such as symbol removal or encryption of critical sections are necessary to guard against reverse engineering. While PX4 Autopilot lacks packing and encryption, basic obfuscation is applied by removing internal symbols as observed when disassembling the firmware.

In summary, the open-source UAV software platform, PX4 Autopilot, lacks the implementation of countermeasures to prevent the majority of vulnerabilities identified in Section 3.

Table 5 Security for PX4 Autopilot and MAVLink

Components	Vulnerability	Countermeasure	Implementation	
GPS module	Fabrication, Jamming	Signal strength monitoring	X	
		Discontinuous sensing data detection	X	
	Fabrication	Cross verification with other sensors	X	
		Monitoring satellite signals number/ satellite identification code	X	
		Time interval check	X	
		Time comparison with accurate clock	X	
Sensor module	Fabrication, Jamming	Discontinuous sensing data detection	X	
		Signal strength monitoring	X	
	Fabrication	Cross verification with other sensors	X	
Communication module	Replay attack	Nonce	X	X
		Sequence number	X	△
		Timestamp	O	O
	Jamming	Heartbeat	△	△
	Flooding	IDS on special-purpose board	X	
	Fabrication, Jamming	Packet authentication	X	O
		Packet integrity check	X	O
	Fabrication, Modification, Sniffing	Packet encryption	X	X
	Deauthentication	Secure password & service	-	X
Firmware	Modification	Secure firmware update	X	
		Secure boot	O	
		Remote attestation	X	
	Buffer overflow	Secure coding	X	
		HW & OS supprot for CFI	X	
	Reverse engineering	Firmware obfuscation	X	

5 SOK: A SECURE UAV BASED ON ARM PSA AND PX4 AUTOPILOT

The security analysis in Sections 3 and 4 highlights multiple vulnerabilities affecting both commercial and open-source UAVs. Therefore, there is a critical need to enhance the security of UAVs by implementing countermeasures against these vulnerabilities.

To address the identified security gaps, a robust security framework should be established. The requirements for enhancing the security of UAVs are as follows: First and foremost, obtaining a Root of Trust (RoT) is paramount. A trusted security service that ensures firmware integrity should be provided based on this RoT. Since the firmware is responsible for performing the critical function of flights, ensuring its integrity requires secure storage to safely store keys and credentials. This is essential for supporting security services such as secure boot, remote attestation, and secure firmware updates. Furthermore, cryptographic functions are necessary for authentication and message encryption between the GCS and UAV. Support for Hardware Security Models (HSMs) such as cryptographic accelerators and True Random Number Generator (TRNG) is required to reduce overhead and improve reliability. Moreover, effective implementation of security mechanisms is crucial to mitigate the vulnerabilities identified in Section 4.

To address these challenges, this section proposes a Secure UAV based on ARM PSA and PX4 Autopilot that allows for easier and faster development while offering flexibility to respond to new vulnerabilities. The proposed Secure UAV builds upon prior research that introduced an ARM PSA-based security platform model tailored for IoT devices [24, 34]. In addition, it is designed by customizing the security platform model to align with the general UAV architecture derived in Section 2 to provide compatibility with PX4 Autopilot.

The ARM PSA is designed to protect critical assets of low-power embedded devices, such as keys, user data, and security services, from unauthorized access [31, 32]. It divides the system memory, peripherals, and functions into Secure and Non−secure states to ensure hardware-based execution environment separation. By isolating critical operations from potentially vulnerable processes, the ARM PSA enhances resilience against cyber threats. The ARM PSA is compatible with the ARMv8-M processor series with suitable processors for UAV implementation, including M23, M33, M35P, and M55.

Fig. 8 shows the proposed Secure UAV based on ARM PSA and PX4 Autopilot. The FCB is based on the ARMv8-M processor and includes HSMs such as One-Time Programmable (OTP) memory, cryptographic accelerators, and a TRNG to enhance security. The firmware executed on this hardware is divided into Secure and Non-secure states.

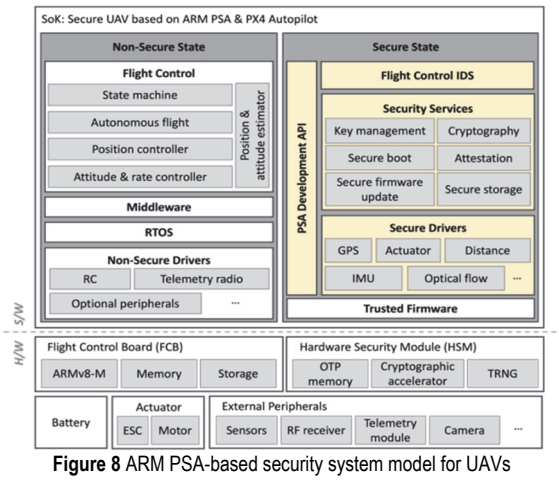

Figure 8 ARM PSA-based security system model for UAVs

The Secure state is based on Trusted firmware and consists of essential features such as security services, secure drivers, flight control intrusion detection systems (IDS), and the PSA development API. The detailed functionalities of each module are as follows:

Secure services: Provide essential functions to ensure the integrity of UAV firmware such as secure boot, attestation, secure firmware update, and secure storage, and the confidentiality of security-sensitive data, such as key management and cryptography. Secure boot verifies the integrity of the firmware during UAV boot-up, and attestation verifies the integrity of the firmware during runtime. Secure firmware update prevents the installation of tampered firmware by validating the firmware image and the provider. Secure storage ensures securely storing of sensitive data such as keys, mission data, and flight logs. Key management facilitates the secure generation, distribution, and revocation of cryptographic keys, while cryptographic services ensure the confidentiality and integrity of UAV communications.

Secure drivers: Support secure data collection and control for UAV sensors essential for flight, such as GPS, actuators, distance sensors, IMUs, and optical flow sensors. By collecting data in a completely isolated execution environment (i.e., Secure state), attacks such as data manipulation through code injection during communication (i.e., Non-secure state) can be prevented.

Flight Control IDS: Prevents external attacks during the communication phase and provides countermeasures. Since most attacks target UAVs through communication, detecting and preventing such attacks as well as ensuring the reliable execution of countermeasures against vulnerabilities are crucial. Therefore, the Flight Control IDS operates in the Secure state. This enables real-time monitoring of control signals and proactive response to anomalies that indicate potential cyber intrusions.

PSA Development API: Provides an interface to easily utilize the secure services offered by the proposed Secure UAV. Developing secure services based on the ARM PSA requires the knowledge of the characteristics of

ARMv8-M processor and the structure of the PSA. However, the proposed model offers APIs that simplify the implementation of essential security features, such as key generation and management, encryption and decryption, signature generation, and secure data storage and loading. This allows for easier and faster implementation of security features with stringent requirements.

The Non−secure state includes non-secure drivers for controlling communication and optional peripherals, a real-time operating system (RTOS) like NuttX for real-time services, middleware for relaying data required for flight control, and Flight Control modules necessary for UAV control. Security-sensitive functions in the Non-secure state can leverage security services from the Secure state via the PSA Development API, ensuring robust protection without compromising system flexibility. This design ensures that the primary flight control and mission-critical operations remain secure while allowing necessary non-secure functionalities to operate efficiently.

The proposed Secure UAV provides a framework for not only to enhance security but also facilitate hardware/software development of UAVs. In addition, the modular design and built-in secure update capability allow for flexible responses to newly discovered vulnerabilities in the future. By adopting this architecture, UAV manufacturers can establish a scalable and adaptable security foundation, reducing development complexity while strengthening overall cybersecurity resilience.

6 CONCLUSION AND FUTURE WORK

As UAVs become increasingly prevalent across various industries, concerns regarding their security vulnerabilities have intensified. However, existing research on UAV security has lacked a comprehensive analysis that considers the software and hardware architecture of UAVs, hindering practical solutions to security issues. To overcome these limitations, this paper presents a systematic analysis of UAV attack cases and proposes a standardized UAV architecture by analyzing commercial UAVs and widely adopted open-source platforms such as PX4 Autopilot and MAVLink. Additionally, this study provides an in-depth security analysis of UAV attack vectors, target components, attack methodologies, and their consequences. Based on the analysis results, a secure UAVs based on ARM PSA and PX4 Autopilot, the proposed security model can provide guidelines for enhancing the security of UAV manufacturing and a framework for easier/faster development of secure UAVs. By systematically addressing UAV security threats, this research establishes a strong foundation for improving the resilience of UAV systems against cyber threats. As a future work, we will focus on implementing and validating the proposed security platform model to assess its feasibility. Furthermore, research efforts will be extended to develop a comprehensive security framework applicable to GCS and UAV communication servers, broadening the scope of UAV security measures.

Acknowledgment

This work was supported in part by the Basic Science Research Program through NRF grant funded by the Ministry of Education under Grant

NRF−2022R1F1A1063724; and in part by the Ministry of Science and ICT (MSIT), South Korea, under the Convergence Security Core Talent Training Business Support Program Supervised by the Institute for Information and Communications Technology Planning and Evaluation (IITP) under Grant

IITP−2024−RS−2023−00266615; This work is the result of commissioned research project supported by the affiliated institute of ETRI (2023-068) and when giving a presentation on this work, the presenter has to clarify that it is the result of the research commissioned by the affiliated institute of ETRI.

7 REFERENCES

[1] Adil, M., Jan, M.A., Liu, Y., Abulkasim, H., Farouk, & A., Song, H. (2023). A systematic survey: Security threats to UAV-Aided IoT applications, taxonomy, current challenges and requirements with future research directions. *Proc. of IEEE Transactions on Intelligent Transportation Systems*. https://doi.org/10.1109/TITS.2022.3220043

[2] Gorman, S., Dreazen, Y. J., & Cole, A. (2009). Insurgents

Hack U.S. Drones. *The Wall Streat Journal*. https://www.wsj.com/articles/SB126102247889095011
[3] Gallagher, S. (2004). Triathlete injured by "hacked" camera drone, *ARS Technica*. https://arstechnica.com/information-technology/2014/04/triathlete-injured-by-hacked-camera-drone
[4] Wlaters, S. *Drone vulnerabilities & attack tools*. https://medium.com/@swalters/how-can-drones-be-hacked-the-updated-list-of-vulnerable-drones-attack-tools-dd2e006d6809
[5] Best, K. L., Schmid, J., Tierney, S., Awan, J., Beyene, N. M., Holliday, M. A., Khan, R., & Lee, K. (2020). How to analyze the cyber threat from drones. *RAND Research Report*. https://www.rand.org/pubs/research_reports/RR2972.html
[6] Charri, L., Chahbani, S., & Rezgui, J. (2003). Vulnerabilities assessment for unmanned aerial vehicles communication systems. *Proc. of International Symposium on Networks, Computers and Communications* (*ISNCC*), IEEE. https://doi.org/10.1109/ISNCC49221.2020.9297293
[7] Yaacoub, J. P., Noura, H., & Salman, O. (2020). Security analysis of drone systems: attacks, limitations, and recommendations. *Elsevier Internet of Things Journal*, *11*, 1-39. https://doi.org/10.1016/j.iot.2020.100218
[8] Chaari, L., Chahbani, S., & Rezgui, J. (2020). MAV-DTLS toward security enhancement of the UAV-GCS communication. *Proc. of Vehicular Techinology Conference* (*VTC2020-Fall*), IEEE. https://doi.org/10.1109/VTC2020-Fall49728.2020.9348584
[9] Jeong, S., Park, E., Seo, K.U., Yoo, J.D., & Kim, H.K. (2021). MUVIDS: False MAVLink injection attack detection in communication for unmanned vehicles. *Proc. of Workshop on Automotive and Autonomous Vehicle Security* (*AutoSec*), NDSS. https://doi.org/10.14722/autosec.2021.23036
[10] Pal, V., Acharya, B. S., Shrivastav, S., Saga, S., Joglekar, A., & Amrutur, B. (2020). PUF based secure framework for hardware and software security of drones. *Proc. of Asian Hardware Oriented Security and Trust Symposium* (*AsianHOST*), IEEE. https://doi.org/10.1109/AsianHOST51057.2020.9358264
[11] Kim, K. & Kang, Y. (2020). Drone security module for UAV data encryption. *Proc of International Conference on Information and Communication Technology Convergence* (*ICTC*), IEEE. https://doi.org/10.1109/ICTC49870.2020.9289387
[12] Allouch, A. Cheikhrouhou, O., Koubâa, A. Khalgui, M., & Abbes, T. (2003). MAVSec: Securing the MAVLink protocol for Ardupilot/PX4 unmanned aerial systems. *Proc. of International Wireless Communications & Mobile Computing conference* (*IWCMC*), IEEE.
[13] Kim, H., Ozman, M. O., Bianchi, A., Celik, Z. B., & Xu, D. (2021). PGFUZZ: Policy-guided fuzzing for robotic vehicles. *Proc. of Network and Distributed Systems Security* (NDSS), NDSS. https://doi.org/10.14722/ndss.2021.24096
[14] Watkins, L., Sartalamacchia, S., Bradt, R., Dhareshwar, K., Bagga, H., Robinson, W. H., & Rubin, A. (2020). Defending against consumer drone privacy attacks: A blueprint for a counter autonomous drone tool. *Proc. of Workshop on Decentralized IoT Systems and Security* (*DISS*), NDSS. https://doi.org/10.14722/diss.2020.23010
[15] Son, Y., Shin, H., Kim, D., Park, Y., Noh, J., Choi, K., Choi, J., & Kim, Y. (2015). Rocking drones with intentional sound noise on gyroscopic sensors. *Proc. of USENIX Security Symposium*, USENIX. https://www.usenix.org/conference/usenixsecurity15/
[16] Jansen, K., Schäfer, M., Moser, D., Lenders, V., Pöpper, C., & Schmitt, J. (2018). Crowd-GPS-Sec: Leveraging crowdsourcing to detect and localize GPS spoofing attacks. *Proc. of IEEE Symposium on Security and Privacy* (*S&P*), IEEE. https://doi.org/10.1109/SP.2018.00012
[17] Kim, T., Kim, C.H., Rhee, J., Fei, F., Tu, Z., Walkup, G., Zhang, X., Deng, X., & Xu, D. (2019). RV Fuzzer: Finding input validation bugs in robotic vehicles through control-guided testing. *Proc. of USENIX Security Symposium*, USENIX. Available from: https://www.usenix.org/conference/usenixsecurity19/
[18] Parrot ANAFI Drone Denial of Service, *Tenable*, 2019 Available from: https://www.tenable.com/security/research/tra-2019-22
[19] Fadilah, M. S. M., Balachandran, V., Loh, P., & Chua, M. (2020). DRAT: A drone attack tool for vulnerability assessment. *Proc. of ACM Conference on Data and Application Security and Privacy* (*CODASPY*), ACM. https://doi.org/DOI: 10.1145/3374664.3379529
[20] Zheng, X. & Sun, H. (2020). Hijacking unmanned aerial vehicle by exploiting civil GPS vulnerabilities using software-defined radio. *Sensors and Materials*, *32*(8), 2729-2743. https://doi.org/10.18494/SAM.2020.2783
[21] Lan, J. K. W. & Lee, F. K. W. (2021). Drone forensics: A case study on DJI mavic air 2. *Proc. of International Conference on Advanced Communication Technology* (*ICACT*), IEEE. https://doi.org/10.23919/ICACT51234.2021.9370578
[22] Pojsomphong, N., Visoottiviseth, V., Sawangphol, W., Khurat, A., Kashihara, S., & Fall, D. (2020). Investigation of drone vulnerability and its countermeasure. *Proc. of Symposium on Computer Applications & Industrial Electronics* (*ISCAIE*), IEEE. https://doi.org/10.1109/ISCAIE47305.2020.9108835
[23] Astaburuaga, I., Lombardi, A., La Torre, B., Hughes, C., & Sengupta, S. (2019). Vulnerability analysis of AR. Drone 2.0, an embedded linux system. *Proc. of Annual Computing and Communication Workshop and Conference* (*CCWC*), IEEE. https://doi.org/10.1109/CCWC.2019.8666464
[24] Jung, J., Kim, B., Cho, J., & Lee, B. (2022). A secure platform model based on ARM platform security architecture for IoT devices. *IEEE Internet of Things Journal*, *9*(7), 5548-5560. https://doi.org/10.1109/JIOT.2021.3109299
[25] DJI Phantom 4 Pro V2.0. https://www.dji.com/phantom-4-pro-
[26] *v2 HoverGames Drone - User Guide and Technical Reference*, (2019). Available from: https://nxp.gitbook.io/hovergames
[27] *Syma* X5C. https://www.symatoys.com/goodshow/x5c-syma-x5cexplorers.html
[28] PX4 *Architectural Overview*. https://docs.px4.io/master/en/concept/architecture.html
[29] MAVLink Developer Guide. https://mavlink.io/en/
[30] *Microsoft Threat Modeling Tool threats*, (2022). https://learn.microsoft.com/enus/azure/security/develop/threatmodeling-tool-threats
[31] Arm Limited, *Arm Platform Security Architecture Security Model 1*.0 Alpha-2, (2019).
[32] https://developer.arm.com//media/Arm%20Developer%20Community/PDF/PSA/DEN0083_PSA_TBSA-M_1.0-bet2.pdf

Contact information:

Beom Seok KIM, Assistant Professor
Department of Artificial Intelligence and Software Engineering,
Major in Information Security,
Chosun University,
South Korea
E-mail: passion0822@khu.ac.kr

Heeseung SON, Student
School of Computing,
Kyung Hee University,
Republic of Korea

Junoh LEE, Student
School of Computing,
Kyung Hee University,
Republic of Korea

Daehee JANG, Assistant Professor
School of Computing,
Kyung Hee University,
Republic of Korea

Deokjin KIM, Researcher
Institute of ETRI (Electronics and Telecommunications Research Institute),
Republic of Korea

Sangwook LEE, Researcher
Institute of ETRI (Electronics and Telecommunications Research Institute),
Republic of Korea

Ben LEE, Professor
School of Electrical Engineering and Computer Science,
Oregon State University,
OR, USA

Jinsung CHO, Professor
(Corresponding author)
School of Computing,
Kyung Hee University,
Republic of Korea
E-mail: chojs@khu.ac.kr