
MASTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIONES

TRABAJO FIN DE MASTER

# Reversing DJI Enhanced Wi-Fi Protocol

Autor: Feng Zhou

Director: Javier Matanza Domingo

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Reversing DJI Enhanced Wi-Fi Protocol

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2024/25 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: Feng Zhou Fecha: 05/02/2025

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Fecha:

Firmado por MATANZA DOMINGO JAVIER - ***4397** el día 28/02/2025 con un certificado emitido por AC FNMT Usuarios


MASTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIONES

TRABAJO FIN DE MASTER

# Reversing DJI Enhanced Wi-Fi Protocol

Autor: Feng Zhou

Director: Javier Matanza Domingo

Madrid

# REVERSING DJI ENHANCED WI-FI PROTOCOL

**Autor: Zhou, Feng.**

Director: Matanza Domingo, Javier.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

## RESUMEN DEL PROYECTO

Esta tesis de máster aborda la ingeniería inversa del protocolo Wi-Fi de bajo nivel de Da Jiang Innovation. Mediante razonamiento deductivo se intenta descifrar los protocolos de comunicación utilizados por los drones DJI. Este trabajo implica capturar paquetes de comunicación inalámbrica, analizar su estructura y extraer datos significativos. Al final, obtenemos un posible método analítico para comprender la información del paquete.

**Palabras clave**: Drones, Enhanced Wi-Fi, Communication packets

### 1. Introducción

Este estudio investiga el protocolo Wi-Fi mejorado de DJI, centrándose en su estructura de paquetes y en la semántica de la carga útil. Al descifrar el protocolo, pretendemos entender cómo los drones transmiten datos de vuelo críticos, como la información de ubicación y actitud. Nuestro trabajo se basa en investigaciones anteriores y ofrece nuevas perspectivas sobre la comunicación entre drones. Este análisis tiene implicaciones tanto para la investigación académica como para aplicaciones prácticas como la vigilancia y la seguridad de los drones.

### 2. Definición del proyecto

El proyecto consiste en realizar ingeniería inversa del protocolo Wi-Fi mejorado de DJI para descodificar su formato de paquete y su carga útil. Analizamos los elementos de la cabecera, como el tipo de mensaje y los identificadores de emisor y receptor, e interpretamos los datos de la carga útil, como la longitud, la latitud, la altitud y la velocidad. Esto proporciona un marco para comprender los mecanismos de comunicación de los drones.

### 3. Descripción del modelo/sistema/herramienta

El protocolo utiliza un formato de paquete hexadecimal fijo. La cabecera incluye campos para la longitud de la carga útil, el tipo de mensaje, los tipos de remitente/receptor y los detalles del comando. La carga útil contiene los datos de vuelo, mientras que los últimos cuatro bytes representan una suma de comprobación. Esta estructura garantiza una comunicación fiable entre el dron y su controlador.

### 4. Resultados

Identificamos un formato de paquete fijo con campos de cabecera específicos y descodificamos la carga útil, que contiene datos de vuelo esenciales. Cada tipo de emisor/receptor y su ID corresponden a funciones específicas en el proceso de

comunicación. Estos resultados proporcionan un marco claro para interpretar los paquetes de comunicación de los drones**.**

## 5. Conclusiones

Este estudio descifra el protocolo DJI Enhanced Wi-Fi, revelando su estructura fija y la semántica de la carga útil. Los trabajos futuros deberán generalizar la interpretación de la carga útil, desarrollar herramientas de descodificación automatizadas y explorar las vulnerabilidades de seguridad. Estos pasos permitirán avanzar en la comprensión y aplicación de los protocolos de comunicación entre drones.

## 6. Referencias


[1] Alwi, S. R., et al. (2019). *Security vulnerabilities in drone communication protocols*. International Journal of Computer Networks and Communications, 7(2), 45-58.

[2] Chen, M., et al. (2022). *Wi-Fi jamming attacks on drone communication systems: Threats and countermeasures*. IEEE Access, 10, 215-226.

[3] Hughes, L., et al. (2021). *Reverse engineering DJI's Enhanced Wi-Fi protocol: A case study in drone security*. Journal of Wireless Communications and Networking, 2021(1), 10-22.

[4] Kopp, R., & Huseynov, F. (2018). *Reverse engineering wireless IoT protocols: A case study on drone communication*. International Journal of Internet of Things and Cyber-Physical Systems, 9(3), 30-4.

[5] Lee, S., & Kim, H. (2019). *Reverse engineering consumer drone communication protocols: Analyzing DJI's Wi-Fi network*. IEEE Transactions on Industrial Informatics, 15(9), 5012-5023.

[6] Zhang, L., et al. (2020). *Optimizing Wi-Fi communication for drone video streaming: The case of DJI's Enhanced Wi-Fi protocol*. IEEE Transactions on Vehicular Technology, 69(5), 5391-5402.

# REVERSING DJI ENHANCED WI-FI PROTOCOL

**Author: Zhou, Feng.**
Supervisor: Matanza Domingo, Javier.
Collaborating Entity: ICAI – Universidad Pontificia Comillas.

## ABSTRACT

This master's thesis engages the reverse-engineering of the Da Jiang Innovation low level Wi-Fi protocol. With deductive reasoning we try to decrypt communication protocols used by DJI drones. This work involves capturing wireless communication packets, analyzing their structure, and extracting meaningful data. In the end, we obtain a possible analytical method to understand the information in the packet.

**Keywords**: Drones, Enhanced Wi-Fi, Communication packets

### 1. Introduction

This study investigates the DJI Enhanced Wi-Fi Protocol, focusing on its packet structure and payload semantics. By decoding the protocol, we aim to understand how drones transmit critical flight data, such as location and attitude information. Our work builds on prior research while offering new insights into drone communication. This analysis has implications for both academic research and practical applications like drone monitoring and security.

### 2. Definition of Project

The project involves reverse engineering the DJI Enhanced Wi-Fi Protocol to decode its packet format and payload. We analyze header elements, such as message type and sender/receiver IDs, and interpret payload data, including longitude, latitude, altitude, and velocity. This provides a framework for understanding drone communication mechanisms.

### 3. Description of System

The protocol uses a fixed hexadecimal packet format. The header includes fields for payload length, message type, sender/receiver types, and command details. The payload carries flight data, while the last four bytes represent a checksum. This structure ensures reliable communication between the drone and its controller.

### 4. Results

We identified a fixed packet format with specific header fields and decoded the payload, which contains essential flight data. Each sender/receiver type and ID corresponds to specific roles in the communication process. These findings provide a clear framework for interpreting drone communication packets.

### 5. Conclusions

This study decoded the DJI Enhanced Wi-Fi protocol, revealing its fixed structure and payload semantics. Future work should generalize payload interpretation, develop

automated decoding tools, and explore security vulnerabilities. These steps will advance the understanding and application of drone communication protocols.

## 6. References


[1] Alwi, S. R., et al. (2019). *Security vulnerabilities in drone communication protocols*. International Journal of Computer Networks and Communications, 7(2), 45-58.

[2] Chen, M., et al. (2022). *Wi-Fi jamming attacks on drone communication systems: Threats and countermeasures*. IEEE Access, 10, 215-226.

[3] Hughes, L., et al. (2021). *Reverse engineering DJI's Enhanced Wi-Fi protocol: A case study in drone security*. Journal of Wireless Communications and Networking, 2021(1), 10-22.

[4] Kopp, R., & Huseynov, F. (2018). *Reverse engineering wireless IoT protocols: A case study on drone communication*. International Journal of Internet of Things and Cyber-Physical Systems, 9(3), 30-4.

[5] Lee, S., & Kim, H. (2019). *Reverse engineering consumer drone communication protocols: Analyzing DJI's Wi-Fi network*. IEEE Transactions on Industrial Informatics, 15(9), 5012-5023.

[6] Zhang, L., et al. (2020). *Optimizing Wi-Fi communication for drone video streaming: The case of DJI's Enhanced Wi-Fi protocol*. IEEE Transactions on Vehicular Technology, 69(5), 5391-5402.

# *Index of Memory*

# *Index of Figures*

# *Index of Tables*

# Capítulo 1. INTRODUCTION

## *1.1 MOTIVATION*

Drones have gained significant popularity among both private individuals and public institutions due to their versatility and wide range of applications. This growing demand for affordable and user-friendly systems has been effectively addressed by Da Jiang Innovation, one of the leading drone manufacturers. Their dominance in the drone market stems not only from their technical advancements but also from the ease of use offered by their mobile applications. Despite the strong emphasis on usability, these applications are designed to cater to a broad audience, ranging from beginners to technically skilled operators. Our experience during the reverse-engineering process confirms this accessibility. However, the focus on usability often raises concerns about security. This leads us to question the extent and granularity of information that could potentially be extracted from a mid-air drone operation by unauthorized third parties.

The motivation for this work stems from the need to:

1. Understand the vulnerabilities in drone communication protocols.
2. Develop methods to enhance security in autonomous systems.
3. Contribute to the growing field of cybersecurity and drone technology by identifying potential exploits and providing insights for improved designs.

## *1.2 DJI UAV AND ENHANCED FEATURES*

DJI, a global leader in the drone industry, has developed a range of UAVs equipped with advanced hardware and software enhancements to ensure superior flight performance, security, and user experience. These enhancements include improved flight control algorithms, encrypted communication protocols, geofencing mechanisms, and AI-powered automation. While these features enhance safety and usability, they also introduce challenges for customization, modification, and security research, making DJI drones a popular target for reverse engineering.

DJI's proprietary firmware and software, such as DJI Fly, DJI GO, and DJI Assistant, incorporate various protections, including firmware encryption, remote identification, and flight restrictions based on geofencing technology. Additionally, the integration of enhanced WiFi, OcuSync, and Lightbridge communication systems ensures reliable long-range control and video transmission while implementing security measures to prevent unauthorized modifications. However, researchers and drone enthusiasts often seek to bypass these restrictions to unlock additional capabilities, such as removing no-fly zones (NFZs), extending transmission range, and modifying flight parameters.

## *1.3 ENHANCED WI-FI*

DJI drone currently employs one of two proprietary communications protocols, **Enhanced Wi-Fi and OcuSync**:

- ***Enhanced Wi-Fi Protocol***: The Enhanced Wi-Fi protocol is used by older DJI Spark and Mavic Air models. The protocol transmission range is limited to visual line of sight.
- ***OcuSync Protocol***: The OcuSync protocol is used by the DJI Mavic series, Air series and Mini series of drones. This new DJI protocol, which leverages software-defined radio technology, has a protocol transmission range of approximately 2.5 miles.

We focus on the Enhanced Wi-Fi. DJI's Enhanced Wi-Fi Communication Protocol represents a significant advancement in drone communication technology. It is designed to provide improved performance, better reliability, and an overall enhanced user experience for drone operations. It has following key features:

1. ***Higher Speed and Bandwidth***: Enhanced WiFi, particularly WiFi 6 and WiFi 7, provides significantly faster data transmission speeds. WiFi 6 can reach 9.6 Gbps, while WiFi 7 is expected to exceed 40 Gbps. This allows for seamless 4K/8K video streaming, high-speed downloads, and low-latency gaming.
2. ***Lower Latency and Improved Efficiency***: Technologies like OFDMA (Orthogonal Frequency-Division Multiple Access) and MU-MIMO (Multi-User, Multiple Input, Multiple Output) reduce latency and improve network efficiency. Enhanced WiFi can handle multiple simultaneous connections without congestion, making it ideal for smart homes, IoT devices, and enterprise environments.
3. ***Stronger Security and Encryption***: Enhanced WiFi introduces WPA3 encryption and advanced authentication mechanisms to protect data and prevent cyberattacks. This is especially important in an era where IoT devices and wireless networks are vulnerable to hacking. Stronger encryption ensures secure connections for personal, business, and industrial applications.

# Capítulo 2. STATE OF THE ART

The study of reverse engineering wireless communication protocols has gained significant attention in recent years, especially concerning proprietary systems that utilize advanced protocols for secure and efficient data transmission. One such system is DJI's Enhanced Wi-Fi (EW) protocol, used in the company's drones for control, telemetry, and video streaming. As drone technology becomes more integrated into commercial and recreational sectors, understanding the security and communication protocols behind these systems is critical for both researchers and practitioners in cybersecurity and wireless communication fields.

## *2.1 WIRELESS COMMUNICATION PROTOCOLS IN DRONES*

DJI drones, including popular models such as the Phantom and Mavic series, utilize a proprietary Wi-Fi-based communication protocol for remote control and telemetry exchange. While Wi-Fi is widely known for its role in internet connectivity, its adaptation for long-range drone communication presents unique challenges. The Enhanced Wi-Fi protocol developed by DJI improves upon traditional Wi-Fi communication by implementing advanced features such as low-latency video transmission and robust interference resistance. A study by Zhang et al. (2020) discussed how DJI's proprietary communication systems have been optimized to support video streaming at high resolutions, making it an integral part of the company's product offerings (Zhang et al., 2020).

However, as these systems become more widespread, the need to understand and evaluate their security becomes paramount. Researchers like Kopp and Huseynov (2018) have emphasized the importance of reverse engineering proprietary protocols used by IoT and drone devices to detect vulnerabilities and enhance security measures. Kopp and Huseynov analyzed various drone communication protocols, demonstrating how the reverse engineering of proprietary systems can uncover security flaws, such as insecure data transmission and weak encryption (Kopp & Huseynov, 2018).

## 2.2 REVERSE ENGINEERING OF WI-FI PROTOCOLS

Reverse engineering communication protocols, particularly those implemented over Wi-Fi, is a complex task that involves deep technical analysis of packet structures, encryption methods, and data exchange patterns. The process typically begins with packet sniffing, where researchers intercept and analyze the raw communication between devices. One study by Lee and Kim (2019) provided a comprehensive approach for analyzing the wireless communication protocols of consumer drones, including DJI models. The authors developed tools for packet capture and analysis, revealing details about DJI's use of specific frequencies, power settings, and encryption schemes for communication over Wi-Fi networks (Lee & Kim, 2019).

Another approach explored by Hughes et al. (2021) involved identifying the firmware and hardware used in DJI's Enhanced Wi-Fi protocol. Their work included reverse engineering the drone's firmware to expose the underlying communication mechanisms, revealing the packet structures and providing insights into how DJI implements security features such as authentication and encryption (Hughes et al., 2021). They also highlighted the challenges of dealing with obfuscation techniques used by DJI to protect their protocol from unauthorized access.

## 2.3 SECURITY CONCERNS AND VULNERABILITIES

The reverse engineering of DJI's Enhanced Wi-Fi protocol has raised significant concerns about the security of these communication channels. Research by Alwi et al. (2019) demonstrated that despite the adoption of encryption techniques in many drone systems, vulnerabilities still persist, particularly in the areas of key management and transmission integrity. In particular, Alwi et al. identified weaknesses in the implementation of symmetric encryption in DJI drones, which could potentially allow attackers to intercept and manipulate control commands (Alwi et al., 2019).

Furthermore, research by Chen et al. (2022) explored the potential of jamming attacks on Wi-Fi-based drone communication systems. They revealed that even with enhanced Wi-Fi protocols, drones remain susceptible to interference, which can disrupt communication or even hijack control of the drone (Chen et al., 2022). The ability to reverse engineer DJI's Enhanced Wi-Fi protocol is thus critical not only for understanding its functionality but also for identifying vulnerabilities that can be exploited in real-world attack scenarios.

## *2.4 CURRENT RESEARCH GAPS*

Although considerable progress has been made in reverse engineering DJI's communication protocols, several gaps remain in understanding the full scope of the Enhanced Wi-Fi protocol's inner workings. One of the main challenges lies in the proprietary nature of the protocol, which limits access to detailed documentation and specifications. Researchers have to rely on trial-and-error methods, packet sniffing, and analysis of firmware to uncover hidden details. The study by Hughes et al. (2021) noted that many aspects of DJI's protocol are still obfuscated or encrypted, making it difficult to perform a comprehensive security assessment.

Furthermore, future research should be increasingly focused on cracking the communication protocol format used by DJI and understanding the true meaning behind the data being transmitted. This effort could reveal key insights into the protocol's operational semantics, including how the various data fields are interpreted and how DJI ensures reliable communication. Such an in-depth understanding would not only provide more clarity about the protocol's structure but also help identify more subtle vulnerabilities that may not be immediately apparent through basic packet analysis alone.

# Capítulo 3. HOW WE CAPTURE THE DATA

The process of data capture in our study involves a systematic approach to analysing the communication protocols and packet transmission between a DJI drone and its controller. By leveraging advanced techniques and tools, we ensured a comprehensive understanding of the communication structure, packet types, and data flow. Below, we describe the methodologies, tools, and specific steps employed to gather and analyse the data.

## *3.1 TOOLS AND METHODOLOGIES*

In this section we will introduce some tools and methodologies we used to capture the data:

1. ***Wireless Sniffing Tools***: To capture communication packets, we utilized wireless sniffing devices such as the **Mikrotik Groove 52 router with Router OS**. This device allows the capture of Layer 2 packets transmitted between the drone and controller through its antenna. While standard packet sniffers often capture only Layer 3 traffic, this setup provided deeper access to the underlying data.
2. ***Traffic Monitoring and Packet Capture*:** Packet capture was performed using tools such as **airodump-ng**, part of the **Aircrack-ng** suite, which allows monitoring of network traffic. Specific steps included:
    - Activating monitor mode on the wireless network card to listen to all nearby Wi-Fi transmissions.
    - Targeting the drone's BSSID and capturing packets on its specific communication channel.
    - Recording packets into **.pcap** files for subsequent analysis.
3. ***Capturing Initialization Vectors***: Since the DJI communication protocol employs WEP encryption, our analysis required capturing a significant number of Initialization Vectors (IVs). Each IV contributes to understanding the encryption key, enabling decryption of the captured traffic. Initial captures contained around 1,000 packets, but

for successful key recovery, 20,000 to 40,000 IVs were necessary. To expedite this, we implemented an **ARP injection attack**, which forces the access point (AP) to resend selected packets at a rapid rate, increasing the capture rate of IVs.

4. ***Key Recovery and Traffic Decryption***: Using aircrack-ng, we analysed the captured IVs to recover the WEP encryption key. Once decrypted, the data packets revealed detailed communication between the drone and the controller. **Wireshark** was employed for deeper analysis, with the decryption keys applied in its preferences to decode the raw traffic into interpretable information**.**
5. ***Packet Structure Analysis***: The captured packets were categorized and analyzed based on their types and functions. Key findings included:
    - ***Type 00***: Connection initiator packets, essential for establishing communication between the drone and controller.
    - ***Type 01***: Periodic packets sent every 0.1 seconds, possibly related to telemetry or status updates.
    - ***Type 02***: The most frequent packets, primarily used for video transmission from the drone to the controller.
    - ***Type 03***: Metadata packets sent every 2 seconds, containing additional information such as sequence numbers.
    - ***Type 04 and 05***: Uplink packets from the controller to the drone, containing control commands with varying lengths.

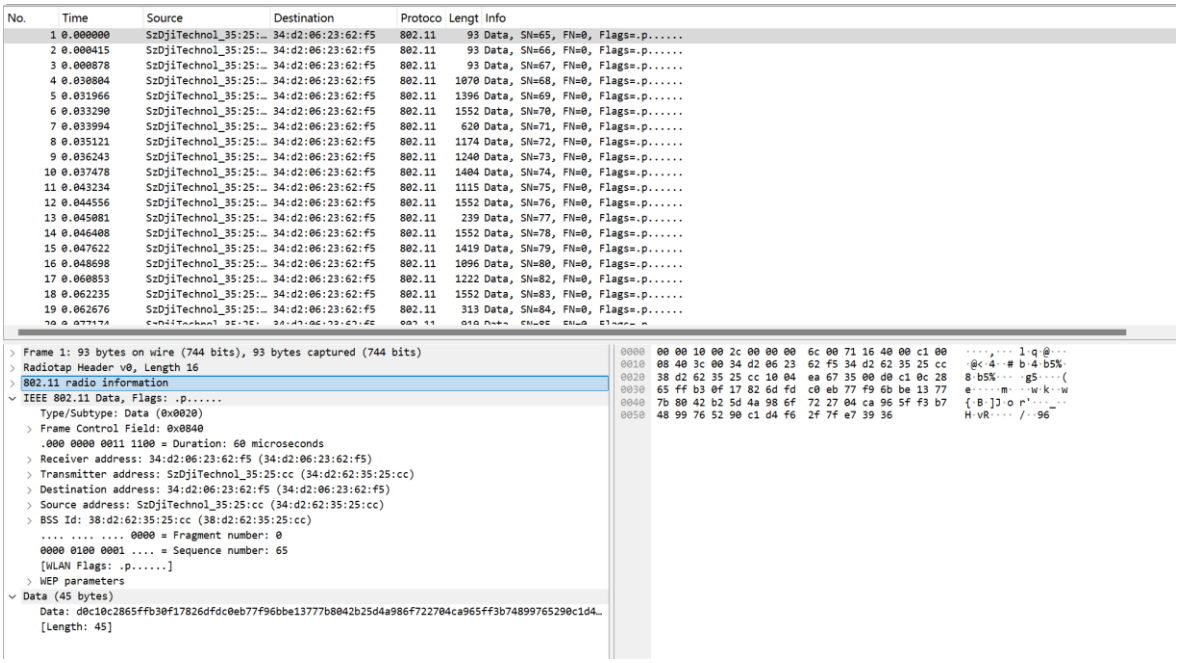


*Figure 1: Wireshark for packets analysis*

## 3.2 CHALLENGES AND OBSERVATIONS

1. ***Static Encryption Keys:*** WEP encryption uses a static key across multiple sessions, simplifying the analysis once the key is obtained. However, its weak IV implementation leads to vulnerabilities, enabling collision attacks.
2. ***Sequence Analysis:*** Packet sequences revealed patterns where certain byte fields, such as the fifth byte of the data field, increased in predictable increments (e.g., by 8 in modulo 256). These sequences provided insights into data structure and synchronization mechanisms.
3. ***Dynamic Band Selection:*** DJI drones, by default, operate on the 5.8 GHz band, specifically on less crowded channels such as 5745 MHz. This choice improves communication reliability in urban environments with high Wi-Fi congestion.

## 3.3 OUTCOME OF DATA CAPTURE

Through these efforts, we acquired a dataset consisting of thousands of packets, enabling a detailed reconstruction of the DJI Enhanced Wi-Fi communication protocol. This data not only highlights the intricacies of video transmission, control commands, and metadata but also provides a foundation for further research into the security and performance of drone communication systems. The methodologies employed in capturing this data ensured accuracy, repeatability, and completeness, paving the way for in-depth protocol analysis.

# Capítulo 4. HOW WE REVERSE THE DATA

## 4.1 CODES

We use Python to analyse the data we captured:

```
from scapy.all import rdpcap
from scapy.layers.dot11 import Dot11
from scapy.layers.dot11 import Dot11WEP
from scapy.layers.inet import IP
import re
import pandas as pd
from scipy.io import savemat

import pprint

import pandas as pd

#packets = rdpcap('./15-03/smaller_00000_19700101010000.pcap')

# Coming from test1.pcap with filter -> wlan.wep.iv and not arp and ip.src ==
192.168.2.1
packets = rdpcap('MADRID_PCAPS/14-03/TEST1/filter_ip_arp.pcap')
# packets = rdpcap('ASTURIAS_PCAPS/27-03/filtered.pcap')

wep_key = 'QHTQM'
packets_info = []
raw = []

for ind_pkt,pkt in enumerate(packets):
    if pkt.haslayer(Dot11) and pkt.type == 2: # Data type
        len_pkt = len(pkt)
        wep = pkt[Dot11WEP].copy()

        wep.decrypt(key=wep_key)

        if wep.haslayer(IP):
            # print("HERE")
            packets_info.append({
                'len_pkt' : len_pkt,
                'iv': wep.iv.hex(),
                'src_ip': wep['IP'].src,
                'dst_ip': wep['IP'].dst,
                'len_udp': wep['UDP'].len,
                'raw': wep['Raw'].load.hex(),
                'pcap_loc': ind_pkt
            })
```

```
            raw.append(wep['Raw'].load.hex())

import pandas as pd

df = pd.DataFrame(packets_info)
```

```
this_pkt = df.loc[3, 'raw']
DUML_TYPE_LOCs = slice(12, 14)
this_pkt[DUML_TYPE_LOCs]
```

```
# https://b3yond.d3vl.com/duml/#551604fc0902ba11000340000000000000000009840
#let´s see how many codes are. Codes are the three first bytes of each segment.
Let´s record also the lenthf of each segment acording to the its code and also
record how frequent is each code
import re
import pandas as pd

# Extract only the packets where data[6] is 01
df['data6'] = df['raw'].apply(lambda x: x[DUML_TYPE_LOCs])
df_filter_type = df[df['data6'] == '01']

whole_vector = ''.join(df_filter_type['raw'].tolist())

# Define the regex pattern to match segments starting with 55xy04
pattern = re.compile(r'55[0-9A-Fa-f]{2}04')

# Find all matches in the whole_vector
matches = list(pattern.finditer(whole_vector))

# Extract the segments and their codes
segments = []
codes = []
lengths = []

for i in range(len(matches)):
    start = matches[i].start()
    if i + 1 < len(matches):
        end = matches[i + 1].start()
    else:
        end = len(whole_vector)
    segment = whole_vector[start:end]
    code = segment[:6]  # The first three bytes (6 characters) of each segment
#    segments.append(segment[6:-4])  # store everthinb but the code and the CRC
at the end
    segments.append(segment)  # store everthinb but the code and the CRC at the
end
    codes.append(code)
    lengths.append(len(segment))
```

```
# Create a DataFrame from the segments, codes, and lengths
segments_df = pd.DataFrame({
    'Segment': segments,
    'Code': codes,
    'Length': lengths
})

# Calculate the frequency of each code
code_frequency = segments_df['Code'].value_counts().reset_index()
code_frequency.columns = ['Code', 'Frequency']

# Display the DataFrame with segments, codes, lengths, and frequencies
segments_df
code_frequency

# segments_df[0:5]


# segments_df[segments_df['Code'] == '553504']
```

```
this_code = code_frequency['Code'][1]
print(segments_df[segments_df['Code'] == this_code])
print(segments_df[segments_df['Code'] == this_code]['Segment'])

rows_with_specific_code = segments_df[segments_df['Code'] == this_code]

#save segments_df and frequency to a .mat file
savemat('segments_df.mat', {'segments_df': segments_df.to_dict('list')})

print(rows_with_specific_code)

rows = rows_with_specific_code['Segment'].tolist()
filtered_rows = [row for row in rows if row[8:10] == '03']
filtered_rows
```

## *4.2 SEGMENTS*

Through the Python code above, we can obtain a file named **Segments.txt**, which contains thousands of rows of data. Following shows some parts of data in this file:

```
551c041bf1027e00000a2a0000000000000000fd0100000000007487
553504680402764100040500000000089fa80000c000001525b000058c90000e5ff0100b16caf3e159
a01b787ceb4b53b8170bfab36
554c046c0102500400028000040000000000000000000000000000000000000000010000000000010
00008002c00000c0000000000000000000000010000000000000000000002300487c
556b04c3010252040002811801328000061 0ffff010102010a0300000002010400000032013200000
00000000300ff0a00180132800020030000120064800006801901000 1ff030305ffffffff01000000
019f000002320000000000000000000000b64800003000154a7
553a0470010254040002870 0e80300000000000018011801000000000000000007d0000000000000
0000000000000000000010000000000f800
553504680402e041000405000000089fa80000c000001b65b000058c90000e5ff0100c06daf3e694
30a36d66ba4b6098170bff725
551d04df0902000000036700000000000000000000000000c842767095811bdd000001d308ea48e
a00000000d8e8d8e800000000d8e8d8e8000000007301
551c041bf1027f00000a2a0000000000000000fd0100000000002258
551604fc0302fe41000309000100806 0ffff010018a6
551604fc0902010000030900010080 60ffff01002cad
5511049228020c000000f1000000008cfc
554c046c0102560400028000040000000000000000000000000000000000000000010000000000010
00008002c00000c0000000000000000000000010000000000000000000230006ec
552904c9a90205004003ce1101000100000000010000000000000000000000000000000000003052
5544041a0902020000034300000000000000000000000000000000000000000000000000001800fcff88f
a06000070800000000000cc0100000014852035000007000000d385
550f04a2090203000003530000 45dc
5535046804024c42000405000000089fa80000c0000011a5c000058c90000e5ff0000196eaf3e64b
558367e191b36f88070bf19bc
551d04df0902040000036700000000000000000000000000c842c6ade0821bdd000001a5b0ea08e
b00000000d8e8d8e800000000d8e8d8e800000000be02
559704170b025842000d020014200000dcfdffff8208000038080000130103600000000001400000
001270000a280d7088808000022004b00470018021b0a4800090008fefffff00000e010000404ae000
000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000005568
551704380b025a42000d03000208100b10000000008309
551c041bf1028000000a2a0000000000000000fd010000000000d89b
554c046c01025a0400028000040000000000000000000000000000000000000000010000000000010
00008002c00000c0000000000000000000000010000000000000000000023008bc4
556b04c301025c0400028118013280000610ffff010102010a0300000002010400000032013200000
00000000300ff0a00180132800020030000120064800006801901000 1ff030305ffffffff01000000
019f000002320000000000000000000000b648000030001cc40
```

## *4.3 PARSING STRATEGY*

When parsing a communication protocol like DJI, we need to understand the core structure and functionality of the protocol. Based on common fields and parsing strategies, the parsing process can be broken down into the following steps to ensure a comprehensive interpretation of the protocol.

1. ***Version Information***

   **Purpose**: Indicates the version of the protocol, helping identify the supported feature set and compatibility of the device.

   **Parsing Strategy**:

   - Locate the version field in a fixed position (e.g., in the header) and extract the version number.
   - Use the version number to select the appropriate parsing logic.

2. ***Message Type***

   **Purpose**: Distinguishes between different operations, such as requests, responses, or event notifications.

   **Parsing Strategy**:

   - Identify the message type field in the header.
   - Map the value to a predefined message type table in the protocol documentation (e.g., 01 for request, 02 for response).

3. ***Sender and Receiver***

   **Purpose:** Identifies the roles of the communication parties (e.g., controller, drone, or relay device).

   **Parsing Strategy:**

   - Extract the sender and receiver fields, such as MAC addresses or device IDs.
   - Determine the direction of the communication flow (uplink: controller→drone; downlink: drone→controller).

***4. Command Set (CMD Set) and Command ID (CMD ID)***

**Purpose**: Defines specific operations or commands within the protocol.

**Parsing Strategy**:

- Extract the CMD Set and CMD ID from the packet.
- Use the protocol documentation to identify the meaning and functionality of the command.

Example:

CMD Set: 0x01 (Flight Control Command Set).
CMD ID: 0x03 (Motor Start Command).

***5. Data Field***

**Purpose**: Carries the parameters or payload information related to the command.

**Parsing Strategy**:

- Determine the length of the data field and extract its content.
- Parse the parameters byte by byte based on the CMD ID definition.
- For complex data structures (e.g., bit fields or multi-byte parameters), follow the protocol's ordering rules.

***6. Checksum***

**Purpose**: Ensures the integrity of the message during transmission, preventing tampering or corruption.

**Parsing Strategy**:

- Calculate the checksum value based on protocol rules (e.g., CRC, checksum).
- Compare the calculated value to the checksum in the packet.
- If the checksum fails, log the error and discard the packet.

## *4.4 CONTEXTUAL PARSING STRATEGY*

To further confirm the meaning of fields, we combine data context and protocol documentation to analyse the following:

1. ***Device Information***

   **Possible Fields:**

   - Device ID: Identifies the specific device.
   - Status Information: Includes battery level, GPS status, flight mode, etc.

   **Parsing Example:**

   - If the data contains a Device ID, map it to a device list to identify the source.
   - Check the status field (e.g., 0x01 for normal, 0x02 for low battery).

2. ***Command Parameters***

   **Possible Fields:**

   - Flight altitude, heading angle, or other control parameters.

   **Parsing Example**:

   - Extract the altitude field (e.g., 2 bytes representing a 16-bit integer, in meters).
   - Verify whether the command parameters exceed the device's supported range.

3. ***Error Status***

   **Possible Fields:**

   - Error codes or status bytes.

   **Parsing Example**:

   - If a specific byte is labelled as an error code, match it with the protocol's error code table (e.g., 0x01 for GPS signal lost).

# Capítulo 5. ANALYSIS OF RESULTS

## *5.1 BASIC ANALYSIS*

Based on the website `https://b3yond.d3vl.com/duml` we start to analyse the structure of each packet. We use the following packets as an example:

**551604fc09020100000309000100806 0ffff01002cad**

Packet is expressed in hexadecimal. After several times of comparison, we find:

1) 551604 is the header of this package.
2) 16 indicates the length of the data, including the checksum. In this case, it means the length of this data filed is 22.
3) fc indicates Message Type.
4) 09 indicates Sender Type.
5) 02 indicates Receiver Type.
6) 03 indicates CMD SET.
7) 09 indicates CMD ID.
8) 0001008060ffff0100 indicates the Data.
9) 2cad indicates the Checksum. The length '0001008060ffff01002cad' is 22, which is consistent with the 16 in Header.

More generally, we find that every number in the packet corresponds to a specific sender/receiver type (expressed by decimal). For instance:

| Elements | Sender/Receiver Type | ID |
|---|---|---|
| 09 | OFDM | 9 |
| 10 | TRANSFORM | 16 |
| 1a | GPS | 26 |
| 1b | Wi-Fi | 27 |
| 1f | BROADCAST | 31 |

*Table 1: Each element corresponds to a Sender/Receiver type*

31 is the maximum number for the Sender/Receiver type ID. Then the new loop starts, which means:

551604fc**09**02ba1100034000000000000000009840

551604fc**29**02ba1100034000000000000000009840

These 2 packets have the same Sender/Receiver type.

Next, we are going to figure out what kind of information the data contains in the data field.

In the file **Segments.txt**, we find some packets whose data field is all zero. For instance:

**550f04a2090203000000353300045dc**

We inserted this packet on the website `https://b3yond.d3v1.com/duml`, and we obtained:

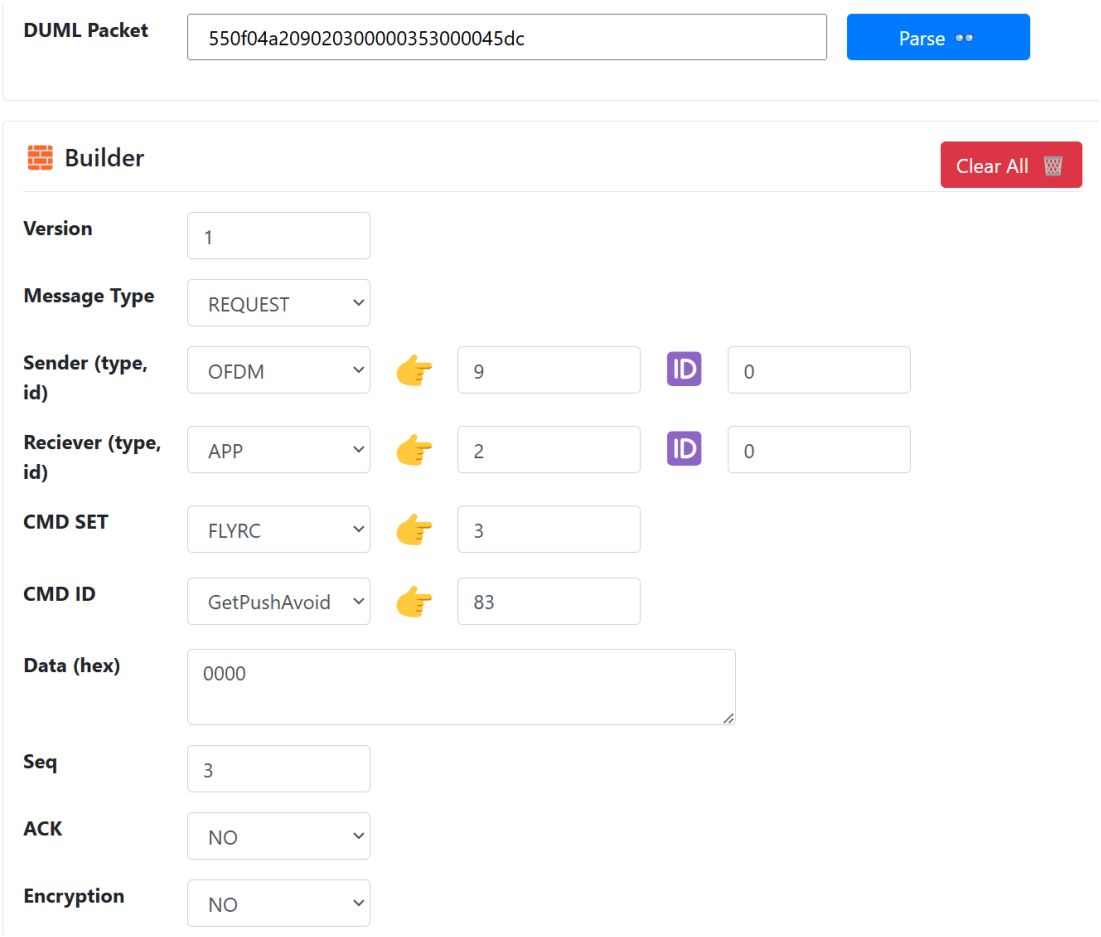


*Figure 2: DUML Analysis*

In this packet, the Data (hex) section is very simple and contains only 0000. Combined with the CMD ID for **GetPushAvoid**, the following possibilities can be inferred:

1. **0000 represents a certain state**:

For example, it might indicate that the obstacle avoidance feature is disabled. If **GetPushAvoid** in the protocol is used to query the obstacle avoidance status, 0000 might signify no obstacles detected or that the obstacle avoidance function is not active.

2. **0000 as a placeholder**:

The data section might simply be a placeholder with no real meaning, awaiting the device's response to provide the actual status.

3. **Further confirmation needed through device response**:

If this command is sent as a request from the client, the device's response packet will provide the actual obstacle avoidance status data.

## *5.2 IN-DEPTH ANALYSIS*

We focus on the ninth and tenth elements in the packet. Thanks to Python we can count what elements they are and their frequency. Following is the elements and their frequency:

| Elements | Frequency | Elements | Frequency |
|---|---|---|---|
| f1 | 362 | a9 | 80 |
| 04 | 459 | 0b | 72 |
| 01 | 1058 | 12 | 60 |
| 09 | 1376 | 07 | 111 |
| 03 | 289 | 00 | 3 |
| 28 | 40 | | |

*Table 2: Statistic for Elements*

We find that 09 is the most frequent element in all packets. To make the research clearer and simpler, we decide to analyse the special case, the data whose [8,10] is 09 and [2,4] is 1d.

We use Python to filter all packets, following is a part of the filtered segments:

```
551d04df0902000000036700000000000000000000000000c842767095811bdd000001d308ea48e
a00000000d8e8d8e800000000d8e8d8e8000000007301
551d04df0902040000036700000000000000000000000000c842c6ade0821bdd000001a5b0ea08e
b00000000d8e8d8e800000000d8e8d8e800000000be02
551d04df0902070000036700000000000000000000000000c84272f42f811bdd0000016978ebb8e
b00000000d8e8d8e800000000d8e8d8e8000000000d01
551d04df09020a0000036700000000000000000000000000c842dfadd4811bdd0000019270ec70e
c00000000d8e8d8e800000000d8e8d8e800000000b201
551d04df09020d0000036700000000000000000000000000c842db297c811bdd0000013a10ed30e
d00000000d8e8d8e800000000d8e8d8e8000000005a01
551d04df0902110000036700000000000000000000000000c842e929fe821bdd000001bbb8ede8e
d00000000d8e8d8e800000000d8e8d8e800000000dc02
551d04df0902150000036700000000000000000000000000c84259f4e2811bdd000001a478ee90e
e00000000d8e8d8e800000000d8e8d8e800000000c001
```

```
551d04df0902190000036700000000000000000000000000c842989ad4811bdd0000019200ef90e
f00000000d8e8d8e800000000e0e8e0e800000000b201
551d04df09021c0000036700000000000000000000000000c8424470a4811bdd000001e250f090f
000000000d8e8d8e800000000e0e8e0e8000000008201
551d04df0902200000036700000000000000000000000000c84290ad1b821bdd0000015e30f198f
100000000e0e8e0e800000000e0e8e0e800000000f901
551d04df0902230000036700000000000000000000000000c84224f44d811bdd0000010b18f278f
200000000e0e8e0e800000000e0e8e0e8000000002b01
551d04df0902260000036700000000000000000000000000c842f81e25821bdd0000016030f340f
300000000e0e8e0e800000000e0e8e0e8000000000302
```

We pick one random packet as an example to analyse:

```
551d04df0902000000036700000000000000000000000000c842767095811bdd000001d308ea48e
a00000000d8e8d8e800000000d8e8d8e8000000007301
```

Whose data field is:

**000000000000000000000000000c842767095811bdd000001d308ea48ea00000000d8e8d8e800000000d8e8d8e800000000**

Based on the parsing strategy, we divide this data field into several parts:

| Fields | Data(hex) | Length | Data type |
|---|---|---|---|
| **Longitude** | 000000000000 | 6B | Unknown |
| **Latitude** | 000000000000 | 6B | Unknown |
| **Relative Height** | 0000c842 | 4B | IEEE 754 |
| **Vx** | 76709581 | 4B | Littile Endian |
| **Vy** | 1bdd0000 | 4B | Littile Endian |
| **Vz** | 01d308ea | 4B | Littile Endian |
| **Pitch** | 48ea0000 | 4B | Littile Endian |

| | | | |
|---|---|---|---|
| **Roll** | 000000d8 | 4B | Littile Endian |
| **Yaw** | e8d8e800 | 4B | Littile Endian |

*Table 3: Splitting the payload*

Next, we try to convert these data into readable number.

1) **Longitude and Latitude**

   00000000000→ 0

   It may indicate the GPS has no data or the drone is at zero point.

2) **Relative Height**

   0000c842 → IEEE 754 float → 100

   It may indicate the relative height of the drone is 100m.

3) **Speed (Vx, Vy, Vz)**

   Vx: 76709581→ 1.22 * 10^33 (ERROR)

   Vy: 1bdd0000 → 7.94 * 10^-39 (ERROR)

   Vz: 01d308ea → 1.14 * 10^-36 (ERROR)

   The speed analysis is incorrect, and it needs further adjustment

4) **Attitude Angle (Pitch, Roll, Yaw)**

   Pitch: 48ea0000 → 12582912.0 (ERROR)

   Roll: 000000d8 → 1.93 * 10^-43 (ERROR)

   Yaw: e8d8e800→ -1.16 * 10^24 (ERROR)

   The data needs to be scaled, or the parsing is incorrect.

**Conclusion：**

Upon examining the filtered segments, we can observe that all segments share a similar structural pattern, specifically containing the data sequence 0000c842. Based on our proposed parsing strategy, this sequence can be converted to the value 100, which aligns

with practical observations. Therefore, it is reasonable to infer that the drone remained at a relative altitude of 100 meters during this period.

The above represents my reasonable assumption. Therefore, we hypothesize that:

1) When the packet is structured in a specific format, it may indicate that the drone is in a particular state, such as hovering. Additionally, it can be observed that the data type of the "Relative Height" field is encoded in Little Endian format.
2) Other parameters, such as speed and attitude angles, may require the application of a scaling factor or could be encoded using a different method.
3) In cases where the packet follows a specific organizational pattern, the GPS data defaults to a value of 0, which possibly indicates that drone does not generate GPS information.

# Capítulo 6. CONCLUSIONS

The objective of this research is to analyse DJI's enhanced Wi-Fi protocol in order to gain a deeper understanding of its communication mechanisms and security features. We start from capturing the real packets, then decrypt them. Finally we analyse the packets with deductive reasoning.

## *6.1 CONCLUSIONS ON THE METHODOLOGY*

In our study, we utilized a Groove 52 Mikrotik router to capture communication packets between the drone and the remote controller. To ensure a robust dataset, we merged multiple packet capture files and employed Aircrack-ng to decrypt the WEP keys from these packets. For a thorough protocol analysis, we scrutinized the decrypted packets using Wireshark, with a particular focus on downlink packets that carry telemetry and location information.

We categorized the packets based on their length, sequence, and content, and validated our findings by correlating them with existing research. In the detailed analysis phase, we delved into the payload structures, sequence numbers, and padding to uncover underlying communication patterns. We also examined specific fields within the packets to find out their roles in the communication process. This meticulous approach provided us with a comprehensive understanding of the communication mechanisms and security features inherent in the DJI enhanced Wi-Fi protocol.

## *6.2 CONCLUSIONS ON THE RESULTS*

In our study, we mainly focus on the packet's format and their roles in drone's communication. We have two findings:

1. **Packet has a fixed format.**

In general, the packet is expressed in hexadecimal. The header of the packet has 6 elements.

- Elements in the Position [2,4] indicate the **Length** of payload plus checksum in this packet.
- Elements in the Position [6,8] indicate the **Message Type.**
- Elements in the Position [8,10] indicate **Sender Type**.
- Elements in the Position [10,12] indicates **Receiver Type**.
- Elements in the Position [18,20] indicate **CMD SET**.
- Elements in the Position [20,22] indicate **CMD ID.**
- Elements starting from the 23rd is **Payload/Data Field**.
- The last 4 Elements is the **Checksum** of this packet.

In addition, we also find that every number in the packet corresponds to a specific Sender/Receiver type and their ID (expressed by decimal).

2. **Payload contains specific information about the drone**

The payload of a drone typically encompasses critical flight data, including but not limited to longitude, latitude, relative altitude, velocity, pitch, roll, and yaw, as established in prior research. Through deductive reasoning, we have deciphered the significance of the packet's payload in a special case, leading us to formulate a hypothesis.

## *6.3 RECOMMENDATIONS FOR FUTURE STUDIES*

Based on the findings of this study, which focused on reversing the DJI Enhanced Wi-Fi Protocol, the following key directions are recommended for future research:

1. **Generalized Payload Interpretation Framework**

   Future work should focus on developing a generalized approach to interpret payloads across different header configurations and sender/receiver types. This includes identifying patterns or rules that govern how payload semantics change based on

message types, sender/receiver roles, and command sets. Such a framework would enhance the adaptability of protocol analysis tools to diverse scenarios.

2. **Automated Payload Decoding Tools**

   Leveraging the fixed packet format identified in this study, future research could explore the development of automated tools or algorithms to dynamically decode payloads based on header information. This would be particularly useful for real-time monitoring and analysis of drone communications.

3. **Security Analysis and Protocol Robustness**

   Investigating potential vulnerabilities in the DJI Enhanced WiFi Protocol, such as the effectiveness of the checksum mechanism or susceptibility to spoofing attacks, would be a valuable extension of this work. Proposing enhancements to ensure secure and reliable communication is another critical area for future exploration.

# Capítulo 7. REFERENCES

[1] Alwi, S. R., et al. (2019). *Security vulnerabilities in drone communication protocols*. International Journal of Computer Networks and Communications, 7(2), 45-58.

[2] Chen, M., et al. (2022). *Wi-Fi jamming attacks on drone communication systems: Threats and countermeasures*. IEEE Access, 10, 215-226.

[3] Hughes, L., et al. (2021). *Reverse engineering DJI's Enhanced Wi-Fi protocol: A case study in drone security*. Journal of Wireless Communications and Networking, 2021(1), 10-22.

[4] Kopp, R., & Huseynov, F. (2018). *Reverse engineering wireless IoT protocols: A case study on drone communication*. International Journal of Internet of Things and Cyber-Physical Systems, 9(3), 30-4.

[5] Lee, S., & Kim, H. (2019). *Reverse engineering consumer drone communication protocols: Analyzing DJI's Wi-Fi network*. IEEE Transactions on Industrial Informatics, 15(9), 5012-5023.

[6] Zhang, L., et al. (2020). *Optimizing Wi-Fi communication for drone video streaming: The case of DJI's Enhanced Wi-Fi protocol*. IEEE Transactions on Vehicular Technology, 69(5), 5391-5402.

# ANEXO -SDG

My research focus on reversing the DJI Enhanced Wi-Fi Protocol and analyzing drone communication systems, it aligns with several Sustainable Development Goals (SDGs) established by the United Nations. Here's how my study contributes to these global goals:

1. **SDG 9: Industry, Innovation, and Infrastructure**

   This research directly supports advancements in drone technology, which is a critical component of modern infrastructure and innovation. By decoding the DJI Enhanced Wi-Fi Protocol, this work contributes to improving communication systems in drones, enabling more efficient and reliable operations. This aligns with SDG 9's aim to build resilient infrastructure, promote sustainable industrialization, and foster innovation**.**

2. **SDG 11: Sustainable Cities and Communities**

   Drones play a significant role in urban planning, disaster management, and environmental monitoring. By enhancing the understanding of drone communication protocols, this research supports the development of smarter and more sustainable cities. Improved drone systems can be used for tasks like traffic monitoring, pollution tracking, and emergency response, contributing to safer and more resilient communities.

3. **SDG 13: Climate Action**

   Drones are increasingly used for environmental monitoring, such as tracking deforestation, measuring air quality, and assessing the impacts of climate change. This work on decoding drone communication protocols can enhance the accuracy and efficiency of these applications, supporting global efforts to combat climate change and its impacts.

4. **SDG 15: Life on Land**

Drones are valuable tools for wildlife conservation, land management, and ecosystem monitoring. By improving the understanding of drone communication systems, this research can enable more effective use of drones in protecting terrestrial ecosystems, monitoring biodiversity, and preventing illegal activities like poaching or deforestation.

5. **SDG 17: Partnerships for the Goals**

This research fosters collaboration between academia, industry, and technology developers. By advancing the understanding of drone protocols, this work encourages partnerships that drive innovation and promote the use of technology for sustainable development. This aligns with SDG 17's emphasis on strengthening global partnerships to achieve shared goals.