Securing the Unprotected: Enhancing Heartbeat Messaging for MAVLink UAV Communications

Isabel Hughes, Adriel Pupo, Jenna Wynd, Zachary Thurlow, Connor Ivancik, and Ying Wang
Stevens Institute of Technology, Hoboken, USA
(ihughes, apupo, jwynd, zthurlow, civancik, ywang6)@stevens.edu

***Abstract*—Despite MAVLink's broad adoption and its flexible architecture that encourages vendor-specific customization, the flexibility introduces substantial cybersecurity vulnerabilities. This paper presents a novel security design aimed at reinforcing the MAVLink protocol against these vulnerabilities, focusing particularly on protecting the heartbeat function, which is essential for maintaining communication links yet previously overlooked in security considerations. Focusing on the vulnerability in heartbeat messaging, we propose a multifaceted security approach that includes a detailed vulnerability analysis, development, and simulation of specific countermeasures such as physical controls, advanced encryption, token authentication, and the use of short-lived tokens for dynamic security. Utilizing a simulated flight plan via QGroundControl for our experiments, we rigorously assess the efficacy of these strategies. Our research significantly advances the conversation on UAV cybersecurity, underscoring the necessity for comprehensive security frameworks that address every aspect of communication protocols, even those previously considered non-critical. By enhancing the security of heartbeat messages within the MAVLink protocol, our work establishes a foundation for more secure and reliable UAV operations across various applications, thus bolstering the overall integrity and reliability of these indispensable systems.**

***Index Terms*—communication protocol, UAV, control, heartbeats, security**

I. Introduction

The rapid development of Unmanned Aerial Vehicles (UAVs), has revolutionized various industries, ranging from aerial cinematography to military applications. These UAVs require lightweight communication protocols to maximize their battery life and ensure successful mission execution. Among the open-source protocols on the internet, one of the most popular is MAVLink (Micro Air Vehicle Link), which is an extremely lightweight protocol that can be adapted for several kinds of UAVs.

Even though utilizing a lightweight protocol is desirable, there are major costs that can come along with it. All packets that are sent over the network are not encrypted, so anyone who is eavesdropping would be able to see the packets sent. If sensitive mission data is intercepted or control commands are modified, it could cause serious concern for both civilian and military applications. There are also severe limitations in access controls, as anyone can connect to a UAV utilizing this protocol as long as they know the correct port to send the messages to. This can leave a system vulnerable to unintended or malicious control attempts.

While existing research has explored various countermeasures for vulnerabilities like man-in-the-middle and injection attacks within the MAVLink protocol, many of these issues stem directly from the inherent lack of security surrounding the protocol's HEARTBEAT message. This critical message is used to advertise the existence of a system on the MAVLink network and can act as a gateway for potential attackers. Any malicious actor can exploit this unrestricted access to establish a connection and infiltrate the system, bypassing potential downstream security measures. Recognizing this vulnerability as an initial cause, our research delves into securing the HEARTBEAT message itself, aiming to provide a robust foundation for overall MAVLink communication security. This targeted approach offers a more fundamental solution compared to simply layering on security measures later in the protocol.

The main contribution of this paper includes:

- Compared to existing research efforts that focus on mitigating specific attacks such as injection or man-in-the-middle attacks, our work pinpoints the vulnerability starting point—unsecured connections initiated through the UAV's heartbeat message. This area is overlooked and underexplored in the current state of the art.
- To address this identified vulnerability, we developed an innovative authentication method specifically designed to secure the heartbeat message. This method is carefully engineered to have minimal impact on UAV battery life, ensuring that the solution is practical for real-world applications.
- By integrating our authentication method into the MAVLink protocol, our approach ensures authorized access to establish connections with UAVs, significantly enhancing the security framework for UAV communications.
- Tailored especially for commercial applications, our proposed methodology is further adapted for various levels of complexity and security levels, aiming to elevate the overall cybersecurity posture of UAV operations using the MAVLink protocol.

II. Related Works

There are multiple works where researchers have looked into the vulnerabilities in the MAVLink protocol. In their overview of the MAVLink protocol, researchers from the Robotics and Internet of Things Laboratory of Prince Sultan University covered a multitude of potential attacks that could

affect the protocol, including Confidentiality and privacy attacks, integrity attacks, availability attacks, and Authenticity attacks.

One method for securing the communication network is physical layer security. This would establish a security model on the physical environment and help prevent systems from potential attacks such as jamming or eavesdropping. An example would be adding artificial noise to the transmission information to either confuse the attackers or verify control command format [1]. Another method is to outsource security. This method would remove the heavy operations of, for example, the UAV, and delegate them to other devices that would be more powerful and less constricted. This would help compute expensive operations and reduce energy consumption [2].

Some other mitigation strategies in robotic vehicles are simplex architecture, enforcers, software partitioning, software rejuvenation, software redundancy, and dynamic architectures. The simplex architecture would detect if abnormal conditions are met and would delegate decisions over to a secure controller, even if the secure controller has reduced functionality. Only when conditions are deemed normal, then the switch back would be made to return control. The benefit would be that the secure controller can be much smaller and trustworthy compared to the complex controller [3].

Various security protocols have been presented by other researchers, but each has their faults. A researcher at the Air Force Institute of Technology conducted a vulnerability assessment of the Mavlink Protocol. In this assessment it is shown that APM 2.5 is sufficiently equipped to secure the MAVLink C^2 protocol using NaCl, however, the tested changes do not include securing the heartbeat, leaving it open to be seen by the attacker. (site)

Researchers at Sfax University [4] have developed a new mechanism called MAV-DTLS which aims to address many security vulnerabilities caused by MAVLink. The MAVLink Datagram Transport Layer Security implements a Transmission Control Protocol, which includes a Handshake protocol and Record protocol, on top of the User Datagram Protocol in order to secure communication between the Ground Control Station (GCS) and UAV. The implementation of this mechanism had no drastic impact on the latency and power consumption of MAVLink when using two Laptops that acted as the GCS and UAV. This mechanism was successful in protecting the simulated UAV from GPS Spoofing, Data Integrity, and Denial-of-Service (DoS) attacks. Additionally, other researchers have accomplished this same outcome by incorporating the ChaCha20 algorithm and Navid encryption algorithm [5]. However, the CPU utilization and memory consumption caused by the protection algorithms are major drawbacks to implementation. Furthermore, vulnerabilities to other unstated attacks are a concern and must be further investigated. Although the implementation of encryption algorithms does aid in protecting data transfer between the UAV and GCS, the resource consumption caused by implementing the algorithms creates an unideal solution.

Researchers at Veermata Jijabai Technological Institute and at Cambridge Institute of Technology proposed similar concepts of using a form of key encryption to protect the data transfer between the UAV and GCS. The main difference is that the research from Veermata Jijabai Technological Institute proposed using a Key-stream Cipher Algorithm to transfer data [6] while the researchers at Cambridge Institute of Technology incorporated the SHA-256 key management scheme with the AES-128 block cipher [7]. In both studies, the proposed security mechanisms showed great results with simulated drones and should be further studied using physical drones. However, these solutions target the data transfer between UAV and GCS rather than the heartbeat connection that our paper will discuss. Other researchers from Jeju, Korea have discussed similar security mechanisms but testing was not conducted [8].

A solution using key encryption has been studied at Air University in Islamabad, Pakistan. Two researchers proposed using Diffie-Hellman key encryption along with the ChaCha20 encryption algorithm to securely send information between a UAV and GCS [9]. Using a simulated drone on ArduPilot, the researchers saw extremely positive results as the algorithms successfully created a secure transfer of information while also having CPU consumption and Memory consumption not differ from the original unsecured version of MAVLink. While this solution does provide many significant benefits, testing on non-simulated drones has yet to occur, and the heartbeat connection that our paper looks to address remains insecure.

There are multiple popular encryption algorithms not discussed in this paper, such as DES, Triple DES, and Blowfish. In exploring how to best protect our heartbeat messages from unauthorized access, it became clear how these encryption algorithms were less suitable for our lightweight application. Data indicates that these algorithms require significantly more time to encrypt and decrypt data, especially when compared to AES. The extended encryption time leads to a higher level of computational demand, which is important to minimize in the context of drone communications since efficiency is needed for preserving battery life. Achieving the best possible battery utilization in drones is crucial for dependable performance, meaning the previously mentioned encryption methods are unsuitable. In contrast to other devices powered by batteries, drones can crash due to battery failure, possibly causing serious accidents [10]. The overhead these encryption schemes introduce could cause latency issues and reduced performance even though they perform well as encryption algorithms [11] Drones require lightweight encryption protocols, so adding the computational burden of DES, 3DES or Blowfish makes them unsuitable regardless of their popularity and efficiency.

III. Motivation and Problem Statement

A. *Analysis of Heartbeat Mechanisms*

Heartbeat messages have significant implications across various sectors of the industry and serve as a staple protocol used for monitoring the availability of resources. They serve as a vital signal that ensures that various components are alive, and as heartbeats are designed to be reliable, they allow for

quick reactions to real-time failures and minimize the chances of false alarms that could trigger unnecessary failures [12].

The heartbeat mechanism is versatile as it sees applications in client-server architectures, peer-to-peer networks, and even online multiplayer games among others. Heartbeats serve multiple purposes and are designed specifically for high-availability systems that incur minimal downtime. Moreover, in load balancing, heartbeats monitor server health, influencing traffic distribution based on server response to heartbeat signals.

There is no denying the critical role heartbeats play in ascertaining the operational status of various systems, however, a lack of security protocol in these continuous heartbeat signals means they could potentially be exploited in attacks due to their lack of encryption and authentication. The persistent transmission of unencrypted and unauthenticated heartbeats heightens the risks of DoS attacks, unauthorized data access, and security breaches. Hence, using strong encryption techniques for heartbeat messages is paramount to protect sensitive information and maintain network integrity.

These insights highlight the foundational role of heartbeat messages in ensuring the smooth operation of complex systems across various industry sectors. They underscore the importance of designing and implementing a robust and secure heartbeat mechanism that will support modern, high-availability, and real-time application environments, especially those described in this paper.

B. *HEARTBEAT Mechanisms in MAVLink Protocol*

The HEARTBEAT message consists of six pieces of informatio as shown in Table I: type of vehicle/component, autopilot type/class, mode, system status, MAVLink protocol version, and a custom field. As these messages are unencrypted, anyone within the network would be able to access all this information. With this, an attacker could easily launch a DoS, Spoofing, or Injection attack. Unfortunately, as the heartbeat is regularly broadcast on the network, an attacker could launch an attack without having to analyze the packets.

The main goal of the heartbeat is to allow other components to discover its existence. As there are common ports that are utilized in the MAVLink protocols, an attacker can attempt to make a connection with the UAV via PyMavlink. The example in Fig. 2 shows the code that attempts to connect to the system and waits for a heartbeat. As a heartbeat is broadcast every second, it does not take long to establish this connection. When the_connection.target_system is equal to 1, a successful connection has been made to the MAVLink network and any message can be sent to it. This means that there needs to be protections in this protocol that prevent an attacker from performing an attack like this from receiving a successful heartbeat.

Unfortunately, MAVLink's open-source information is not very extensive on the heartbeat protocol, so there is not much more that we know about it. When looking at the GitHub code, we can see that the network is monitored, and if a new port is discovered that is sending MAVLink packets, it is automatically bound to that new port and starts broadcasting heartbeats to it. In the next section, we will propose different strategies for securing heartbeats to stop vulnerabilities at the entry point.

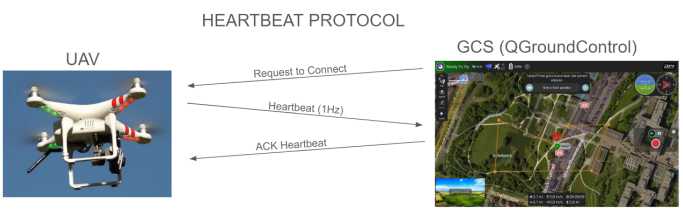

Fig. 1. Current Heartbeat Protocol with GCS

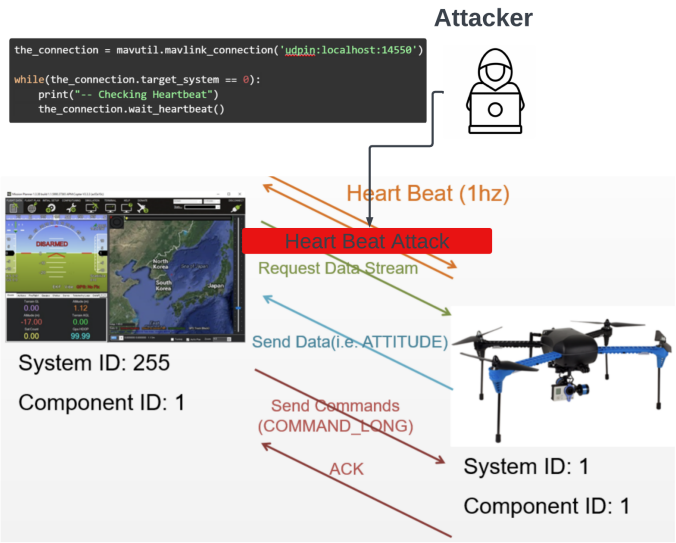

Fig. 2. Heat Beat Attack Intrusion

C. *HEARTBEAT Caused Attack Model*

Although open-source information about MAVLink's heartbeat can not be found, simple experimentation demonstrates how weak the current system is.

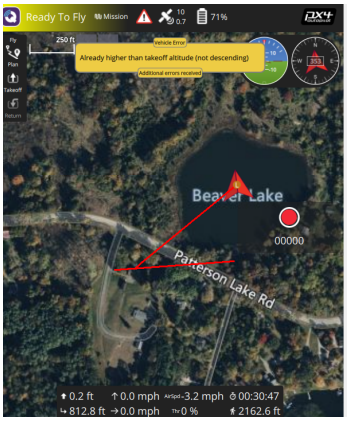

Fig. 3. Successful Injection Attack

TABLE I
STRUCTURE OF HEARTBEAT PACKET FROM MAVLINK

Field Name	Type	Values	Description
type	uint8_t	MAV_TYPE	Vehicle or component type. For a flight controller component the vehicle type (quadrotor, helicopter, etc.). For other components the component type (e.g., camera, gimbal, etc.). This should be used in preference to component id for identifying the component type.
autopilot	uint8_t	MAV_AUTOPILOT	Autopilot type / class. Use MAV_AUTOPILOT_INVALID for components that are not flight controllers.
base_mode	uint8_t	MAV_MODE_FLAG	System mode bitmap.
custom_mode	uint32_t		A bitfield for use for autopilot-specific flags
system_status	uint8_t	MAV_STATE	System status flag.
mavlink_version	uint8_t_mavlink_version		MAVLink version, not writable by user, gets added by protocol because of magic data type: uint8_t mavlink version

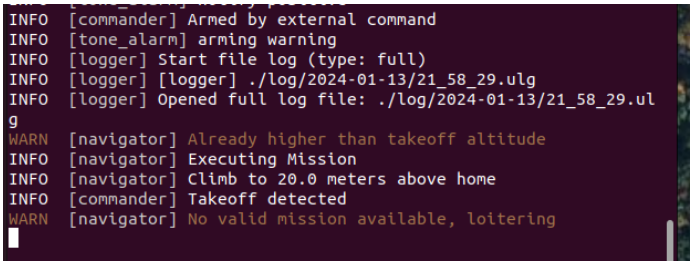

Fig. 4. Message from UAV After Injection Attack

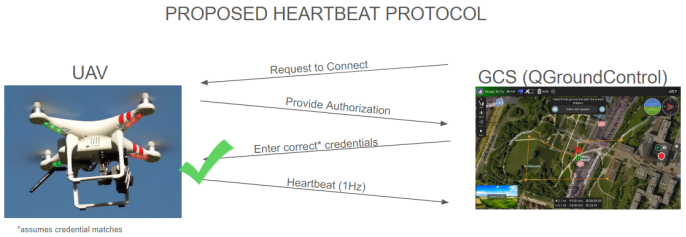

Fig. 5. Passed Heartbeat Protocol with GCS

Fig. 1 shows how the heartbeat communicates with the GCS, moreover, the weakness of the heartbeat is shown when comparing Fig. 1 and Fig. 2. Although the UAV is communicating with an attacker, there is no distinct way the drone can properly differentiate between the GCS and the attacker. This creates an extreme ease of access for an attacker to infiltrate the drone.

In Fig. 5 and 6, we demonstrate the implementation of

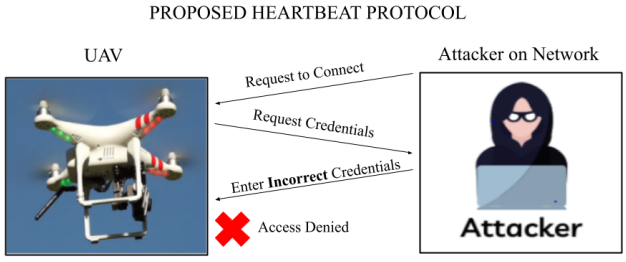

Fig. 6. Rejected Heartbeat Protocol with Attacker

a secure secret key mechanism. In this system, whoever is trying to connect must provide credentials and a secure shared password in order to receive an initial heartbeat from the drone and successfully connect to it. In turn, this prevents any outside attacker from connecting to the drone without knowing the secret password ahead of time. Additionally, this design creates openness for the implementation of different defense mechanisms.

Shown in Fig. 3, we followed the attack model, and we were able to successfully launch an injection attack against the UAV. After starting a flight plan, we launched our injection attack, which causes the UAV to hover in place until it is given more instruction or the battery dies. In Fig. 4, you can see the message from PX4 which states that the drone no longer has a mission and is left hovering in place. From there, we were able to send a new mission and divert the UAV to a new location. Any unauthorized user on the network would be able to launch this simple attack and effectively steal or crash the UAV.

IV. DEFENSE METHODOLOGY DESIGN

While there are several different strategies to prevent unauthorized connections, searching in the realm of authentication and authorization methods may be the answer. This section explores various strategies to ensure that only authorized individuals can establish connections with a UAV.

A. *Encryption Through Secure Handshake Protocols*

Heartbeat message encryption addresses their inherent insecurity as they openly broadcast a present and responding MAVLink component. This simplicity, while beneficial for performance and reliability, leaves them exposed to eavesdropping. Adding a layer of security is crucial in protecting against unauthorized access and maintaining the integrity and confidentiality of communications from the drone. A proposed method [13] combines secure handshake protocols and advanced cryptography techniques to safeguard heartbeat messages between a ground control system and a drone. It starts with both software and SSL handshakes to establish a secure channel. A unique session key is generated using a combination of the drone's permanent identifier and a temporally relevant location ID, which is processed through a key derivation function. This approach of leveraging RSA for

public and private key cryptography ensures each communication session is encrypted with a freshly generated, self-expiring key, enhancing security against interception and unauthorized access.

Before the implementation, there should be some form of software handshake, which will help to control the transmission of the heartbeat message and establish trust between the drone and the control station. After the software handshake, an SSL handshake, used to secure internet communications, establishes a secure channel through symmetric encryption. The combination of the two provides multiple layers of security, therefore enhancing it, as other layers would protect drone operations if one is compromised.

The approach then describes a method for encrypted communication during a session between a drone and a ground control station. Session key generation is done using the drone's unique identifier and a location identification (Location ID) which is determined at the time of key derivation. A Key Derivation Function (KDF) uses this information to produce a session key unique to each session. This ensures that the session key, also known as a symmetric key, is sixteen bytes (a block size of 128 bits) and provides a secure communication channel. The session key is self-expiring after a defined time interval, adding an extra layer of security. The sender will need to provide a new session key if one expires, and after generating the new session key, it will be exchanged between the sender and receiver and utilized for encrypting and decrypting data during the upcoming specified period.

After the generation of a session key, the RSA cryptosystem used for public and private key cryptography in securing drone communications should be used. This involves two primary steps: RSA key exchange and session encryption. The RSA public-key cryptosystem ensures the secure exchange of the session keys generated previously. In this process, the drone encrypts the session key using its public key, ensuring that only the holder of the corresponding private key, the authorized ground control station, can decrypt it. This mechanism guarantees the session key's security, protecting it from being compromised even if the transmission is intercepted by a malicious user attempting to connect to the drone's heartbeat message. Once the session key is securely exchanged and both parties have access to it, they employ this key for symmetric key encryption. This method is used to encrypt and decrypt all communications during the session, providing an efficient and secure means for authorized ground control users to access a drone's heartbeat. This approach not only enhances the security of the data being transmitted but also ensures the integrity and confidentiality of the communication between the drone and the ground control station.

B. Encryption Through Lightweight Protocols

The use of encryption techniques to protect heartbeat messages from drones, while greatly increasing security, presents several challenges. The computational requirements for encryption and decryption schemes increase overall battery consumption and reduce the duration of operation of the drone. This is a particular problem for UAVs designed to handle continuous use, meaning energy efficiency is critical. Furthermore, the challenges of implementing strong encryption protocols make it difficult to integrate with existing systems, regardless of their computational complexity. In particular, changing the sensitive MAVLink heartbeat system to accommodate increased levels of security not only can trigger conflicts but also require significant configuration changes to the heartbeat protocol. Such changes carry the risk of compromising communication in real-time performance and reliability, making system behavior unpredictable and difficult to monitor. It is crucial to integrate an encryption algorithm that is easy to implement, and it's notable that lightweight encryption algorithms, such as PRESENT, SPECK, SIMON, LED, and ASCON, offer a compromise by increasing speed, requiring fewer resources compared to traditional algorithms, and are easy to implement.

These lightweight options are more suitable for drone communications, offering a viable solution to secure communication without compromising battery consumption [14]. Among these, ASCON stands out as a lightweight security protocol known for its robust security and ease of implementation. [15] Selected as a new standard for lightweight cryptography in the NIST Lightweight Cryptography competition (2019–2023), ASCONs design seems particularly suitable for UAV communications because of its lightweight characteristics and numerous features. ASCON can be used to secure the heartbeat message from a drone using its authentication encryption and hashing features, ensuring that the heartbeat has an extra layer of confidentiality and integrity. Any tampering with the message during transit would be detected, preventing anyone from modifying the contents of the heartbeat: type of vehicle/component, autopilot type/class, mode, system status, MAVLink protocol version, etc. The provably secure mode with keyed finalization adds an extra layer of security as well. The encryption and decryption processes do not introduce latency into the drone's communication system and are designed for constrained devices like drones, ensuring that securing the heartbeat messages does not impact drone performance. Furthermore, ASCON can encrypt and decrypt messages in a single pass, without the need for multiple rounds, helping to increase operation speed. ASCONs features ensure that messages sent from the drone remain confidential and tamper-proof, without compromising the drone's performance or operational efficiency.

ASCON mode for authenticated encryption includes multiple stages: initialization, where the encryption process starts using a nonce (an arbitrary number used only once) (IV) and a key (K); processing of Associated Data (A), which can be information that does not need to be encrypted, such as the MAVLink version number sent in a heartbeat; the encryption of the actual Plaintext (P) into Ciphertext (C) across multiple blocks; and Finalization, where the process ensures that the output (T) confirms the integrity and authenticity of the encrypted message.

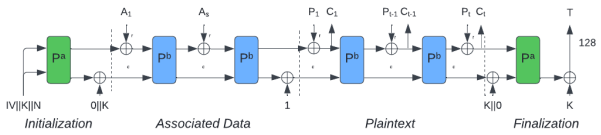

Fig. 7. An Overview of ASCON for Authenticated Encryption

```
Type: 2, Autopilot: 12, Base Mode: 8,
Custom Mode: 0, System Status: 4, Mavlink Version: 2

Encrypted Message: b'\xef|\xb5\x85\x92T\xc1h\xbe\x00\xb8~\x14\r\x89\xe90\xf5d9\x17\xf2Au\x9d'
Decrypted Message (as bytes): b'\x02\x0c\x08\x00\x00\x00\x00\x00\x04\x02'

Deserialized Message:
Type: 2, Autopilot: 12, Base Mode: 8,
Custom Mode: 0, System Status: 4, Mavlink Version: 2
```

Fig. 8. An Example of Encrypting and Decrypting the Heartbeat message using ASCON Encryption Scheme

C. *NFC/RFID Scan*

The idea for putting physical controls on the drone came from analyzing RFID door locks on our campus. This is a fairly simple implementation of a control that could reject any new connections. Before a flight plan is launched, an authorized person would scan a tag, which tells the UAV to not actively search for new devices and only send heartbeats to the currently connected components. Once the mission is completed, this scan can happen again and would allow new connections again. This control method could potentially mitigate unauthorized accesses that utilize the heartbeat vulnerability.

While this is possibly a good method for some security, this is not a complete solution and does not protect against eavesdropping on the network or someone connecting to the UAV before the control is activated. There may also be ways to clone the RFID/NFC tag and bypass this security method. Lastly, the frequency of the scanner could mess with onboard electronics or other nearby devices, and cause further issues.

D. *Biometric Authentication*

Biometrics is the use of a unique physical trait or characteristic as a security measure instead of something you know, such as a password. This could be either fingerprint scanning, voice recognition, or any other physical trait that is specific to one person. It is ”most suitable for personal identification” [16]. This eliminates the possibility of a user forgetting their password but comes with its own set of drawbacks. This method requires added hardware to the drone, something that must be kept as limited as possible as any added weight can affect how much the drone can carry and how long the battery last. It also limits who can use the drone, as any temporary user would need to be established in the system, rather than given a temporary password or key. There also may be issues with the equality of users because some physical limitations, such as not having fingerprints, may limit personnel.

Researchers have doubts about these security measures, however. In a literature review of papers about biometric authentication schemes, a researcher concludes that ”although there has been a lot of development in recognition accuracy of the biometric recognition systems... still there persist various open concerns that researchers require to settle.” [17]. It is possible that the system can produce false positives or mistake different users for each other because of similar physical traits.

E. *Passphrase/Token Authentication*

Similar to the idea for physical controls where there is some setup before the UAV is on its mission is passphrase authentication. Before a mission is started, any connections that want to be made must enter a password. Normal passwords are insecure and brute-forceable, something like a YubiKey or an RSA SecureID token, where the token code is constantly changing. This would be a strong authentication method, especially without the usage of usernames and passwords. Even if a passphrase is compromised, it would be changed frequently and reduce the risk of unauthorized access. A similar option would be to use an authenticator app that works offline, such as Okta or Linux’s Authenticator. These applications use the same method as RSA SecureID tokens, with a six-digit code changing at regular intervals, but has the benefit of being behind the password wall of a smartphone.

Using physical authentication methods like YubiKeys and RSA SecureID tokens offers extra protection against unauthorized access through digital signature verification. There are several options when it comes to choosing a method for 2FA, but the functionality of YubiKeys makes it seem practical for drone use. For example, a Yubikey has centralized management for provisioning and life cycle management, which would allow this method to be used in larger or smaller-scale fleets. In terms of functionality, a key pair that is connected to the certificate on a certain user’s YubiKye is created. This pair comprises a private key that is stored on the YubiKey and shielded from external access, which concerning drone use, makes them an authorized user able to receive heartbeats from the drone to launch a mission protocol. The pair also consists of a public key, which can be distributed openly and safely. The act of digitally signing a heartbeat message involves encrypting the data with a hash and then using the private key to sign this hash, producing a unique digital signature. This signature and the original data can be verified by anyone who has the public key. [18]

Some potential drawbacks of this method are the cost, the risk of losing a token, and the complexity of integration. Having hardware tokens for each drone operator and the cost of the system could potentially make this not feasible for companies with small budgets. Free distributions of authentication tokens exist, which could prevent cost from being a factor. Authenticator apps also require smartphones, this may increase the need for work phones, if they are not already provided, or mandate an agreement with employees to install company encryption and software onto personal devices. There is also the worry of protecting the token itself from being stolen or lost. This is a risk that would have to be factored into utilizing this method. Tokens also work off of an internal clock, and

	Encryption through secure handshake protocols	Encryption through lightweight protocols	NFC/RFID	Biometric	Password/Token Authentication
Easy to Deploy	X		X		X
High Security	X	X			X
Easy to scale			X		X
Takes up significant battery	X			X	
Costly				X	

Fig. 9. Comparison of Methodologies

if they fall behind, they may need to be re-synchronized with the host system. With a central management system, this would not be too difficult to fix. Lastly, integrating these tokens within the MAVLink ecosystem might present unknown complexities. These complexities can cause issues for the battery life of the UAV or issues with the protocol itself. Hopefully, keeping the authentication to only happen before the drone takes off will preserve the battery and ensure the mission is still carried out successfully.

V. Testing & Outcomes

A. *Simulation Configuration and Experiment Design*

During the initial planning stages of the MAVLink exploitation, the simulated flight plan was created using PX4 Autopilot, QGroundControl (QGC), and its integrated waypoints.

However, the internal QGC system did not have the functionalities and consistency that we were looking for. Thus we turned to PyMavLink, a Python library that allows for MAVLink connections between drones and QGC. This connection is done by utilizing the computer's IP network and port, which coincides with the pre-planned assumptions of our exploitation [19]. With the mission plan in place [20], and by designating the drone's latitude and longitude, our drone was successfully able to complete a full flight.

*1) **Comparison of Methodologies**:* Selecting an appropriate authentication method for UAVs necessitates careful consideration of various factors. This comparison focuses on the difficulty level, computational complexity, generated risks, and prior information required for each method.

Encryption offers a strong layer of security by protecting the confidentiality of data but comes with increased complexity and potential computational demands. Utilized algorithms should minimize the overall resources required to have a robust security protocol. Lightweight encryption protocols do not require significant computational power and are known for their scalability, making them potential candidates for drone use. These include protocols previously mentioned: PRESENT, SPECK, SIMON, LED, and most importantly, ASCON, which is the most robust of these protocols. Integrating ASCON into the MAVLink heartbeat protocol would involve the inclusion of the ASCON library to encrypt heartbeat messages at the drone level and decrypt them at the authorized receiving devices. However, based on the performance charts for ASCON C implementations, ASCON shows promise in terms of efficiency and performance for securing messages in MAVLink. The cycles per byte for long messages vary across different processors. For example, 7.8 cycles per byte for encryption with ASCON on an Intel Core i5 processor indicates relatively efficient performance for a cryptographic algorithm [15]. The impact on drone resources should then be minimal.

RFID/NFC scans offer a low difficulty level and require minimal computational resources, but they are difficult to scale. Scaling an RFID/NFC system is not just about the technology itself, but also involves the infrastructure required to support a larger network of readers and tags. For example, a system such as YubiKey may be difficult to scale due to factors such as the cost of each key and keeping track of which key belongs to which drone. They are also relatively cheap to implement within a company respective to cost and computing power. Most scanners have a typical consumption of 100mA. For 2 hours of operation with 80% efficiency, the power draw would be 100mA * 2h * (1/.8) = 250mAh. Compared to the normal battery draw of a UAV, this will not be computationally intensive for the battery life of the UAV. Implementing this method requires knowledge of the target device ID and scanning range. While RFID scans have larger ranges, NFC utilizes 0-20cm ranges. With this taken into consideration and their vulnerability to cloning and spoofing attacks, there is moderate potential for unauthorized access and other security breaches.

While biometric authentication may be more expensive than RFID authentication, it provides enhanced security by analyzing some features (fingerprint/retina/face) of a user. RFID is very easy to bypass and can be by stealing a physical ID card. Biometric systems are extremely accurate and almost impossible to bypass. Capturing the scan itself takes very minimal processing power, alongside feature extraction. Several efficient algorithms exist to process whatever image is captured. The computational complexity rises when matching the scan against registered scans. Especially with a bigger database, the scan must be compared against all the authorized users, and that may increase the processing power utilized to verify one's identity. Adding new users to the system can however slow the process, as ensuring users have the correct tags or biometric scans to be authenticated takes time. Biometric authentication also has a medium risk level, as it brings about privacy concerns and vulnerabilities to spoofing attacks. Implementing this method requires an attacker to enroll their biometric data into the database or somehow access an authorized user's biometric data.

Finally, passphrase/token authentication offers a good balance between security and convenience, especially when utilizing strong passphrases and secure tokens. While some token systems are more expensive, some are free to use with an authorized user's cellphone and eliminate the factor of cost

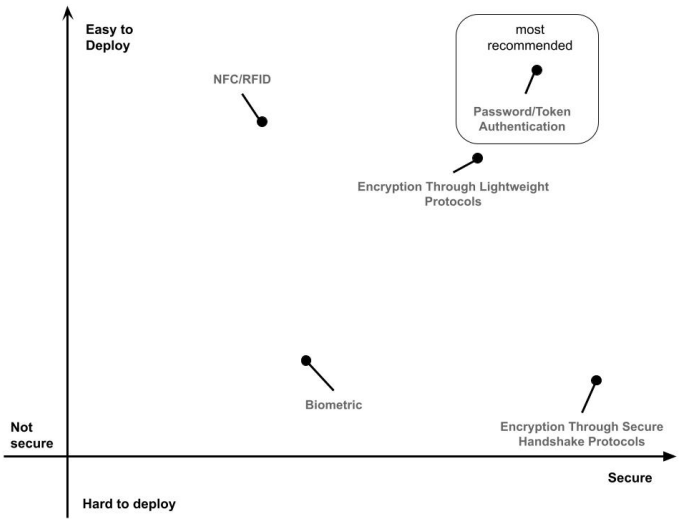

Fig. 10. Comparison of Methodologies

and difficulty of implementation. As these tokens work offline with a known time of start and secret seed, their computational complexity is quite low and the new token value is recalculated every so often using basic mathematical operations. Losing a token or phone could also be a risk, but with a management system, lost devices or tokens can be deactivated and will no longer be usable for authentication. For someone to be authenticated in this method, they would need to know their credentials and have access to the changing values of the physical or software token.

B. *Results Analysis*

Based on all our research, token authentication is shown to be the most suitable method for authentication for the MAVLink Protocol.

*1) **Security**:* While encryption protects data confidentiality, it does not necessarily prevent unauthorized access to the drone itself. Biometrics have the ability to provide for strong authentication but raise privacy concerns around storing large amounts of user biometrics. These databases may also be insecure and an attacker could figure out how to get their biometric scan into the database. RFID/NFC scans, while convenient, are vulnerable to cloning and spoofing. Between the different security concerns, tokens and passphrases offer a good balance of stronger cybersecurity defenses. They require not just having the token but also knowing the current value, which changes frequently. Additionally, tokens can be used for two-factor authentication, adding another layer of security.

*2) **Convenience**:* While encryption doesn't effect usability and convenience, its integration could slow down data transmission rates with the increased processing power that it utilizes. RFID/NFC are very user-friendly, as you just need to tap your tag near the scanner device. This does limit convenience in terms of proximity, as a user would need to be physically close to where the scanner is. There is also the need for having this tag on your person at all times, and brings worries about if it is lost or copied, which is quite easy with a skimmer. Biometrics are relatively quick and user friendly, but can be bothersome to repeat several times if needed. They also may require specific positioning and more expensive technology to ensure the scan does not fail. Tokens are highly convenient, whether software or hardware. Software tokens are readily available on a smartphone, laptop, or other device, which eliminates the need for carrying a physical token. While a hardware token may be less convenient due to the possibility of loss, life-cycle management systems can ensure they are deactivated and not able to be used.

*3) **Resource Efficiency**:* Encryption algorithms can be computationally expensive. While lightweight options exist, encrypting and decrypting data can still drain battery life, especially for real-time communication. The chosen algorithm and processing power of the UAV's onboard computer will significantly impact efficiency. RFID/NFC have low computational needs and require very minimal processing power. However, their overall efficiency in the system depends on the specific reader type. For biometrics, with a larger database, stronger processing power will be required to identify a user. Also depending on the type of scan taken, more processing power may be needed for higher accuracy. Tokens require minimal processing power, as generating codes frequently is based on mathematical operations that can be done offline and extremely fast.

*4) **Implementation of Passphrase/Token Authentication**:* In figures 5 and 6, we showed our new proposed authentication model. MAVLink's Github repository provides us with an example of how the UDP connection is set up between the UAV and the GCS. To ensure that this method would protect against unauthorized users, we made a demo of how this would work. Utilizing C, we made a simple GUI that takes in a user ID and passphrase, along with a token that would be constantly changing. For now, we have the login information and the secret token hard-coded, but in real implementations, this data would not be available in the code. Also, most companies when utilizing this token key system have the digits tacked onto the password, so both pieces of data have to be correct together. For this implementation, the boxes are left separate.

As shown in Fig. 11, when a user attempts to make a connection to the UAV, they are prompted with a pop-up box for authentication. Successfully entering the correct data within three attempts results in a confirmation message, and the UAV establishes a connection with the user's system to start relaying messages. Conversely, entering incorrect credentials prompts an error message. Users have up to three attempts to input the correct credentials. Failing to do so after the third attempt triggers a final message indicating the denial of the connection, and the authentication process is terminated.

While this implementation of our proposed protocol is very preliminary, it significantly enhances security by reducing the potential for unauthorized access compared to the original protocol. For a demonstration of our approach, refer to our

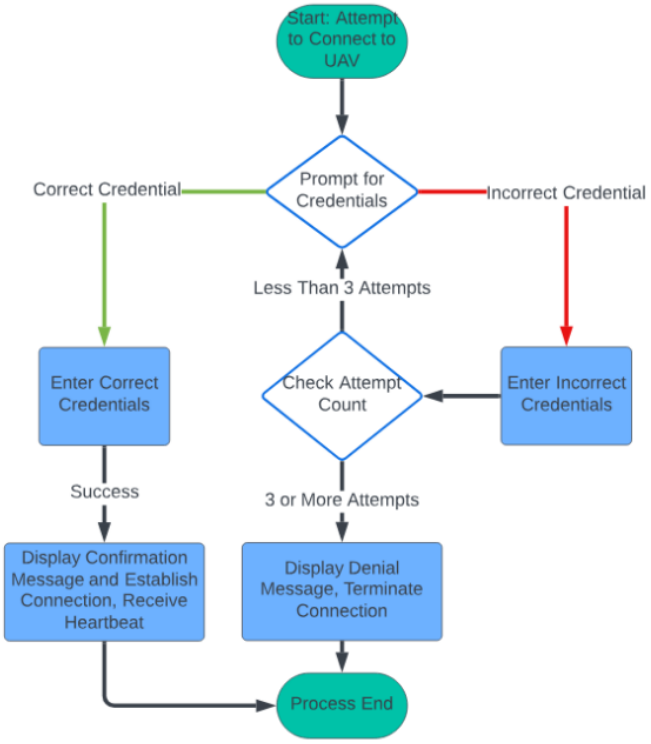

Fig. 11. Flow-graph for Heartbeat Authentication

project's GitHub repository.[1]

When comparing the amount of possible attacks before and after the implementation of this methodology, it can be seen that there may still be a possibility for attacks, but the chances are far less likely. Security tokens utilize a 128-bit seed which means there are about 500 quintillion potential sequences that a token can produce. This makes the token value almost impossible to brute-force. In order to use them reliably, users must protect themselves against social engineering tactics and phishing attacks. If an attacker were to gain access to the token database or an authorized users token, they would only require a users credentials to launch an attack. Since messages would still be unencrypted on the network, there would be a possibility for replay attacks or other eavesdropping attacks which may allow an attacker to grant themselves access for connection. Overall, without this methodology, any attempt to connect to a UAV would be successful. With token and passphrase authentication enabled, most attacks would be blocked, but there may be other attacks which can bypass the authentication. While is difficult to quantify the attack reduction due to the ever evolving cybersecurity threat landscape, we can say the attack threat level has significantly decreased from always being possible, as shown in Fig. 12.

VI. Conclusion

This paper introduces the idea of focusing on the HEARTBEAT to potentially secure the communication between drones

[1]https://github.com/ihughes22/2023-2024-MAVLink-Vulnerability-Detection

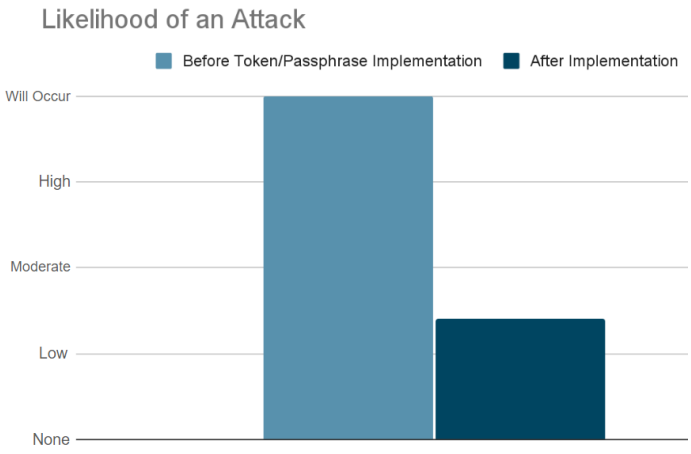

Fig. 12. Likelihood of Attack Before/After Methodology Implementation

and the drone's controller, as the HEARTBEAT is needed to create and uphold a connection. As demonstrated, there is currently a risk of injection attacks on the PyMavlink system that could be resolved with added security. Multiple solutions to this issue are presented, including adding password or token requirements to the connection or encrypting the HEARTBEAT, however, any added requirements must be kept lightweight as the drone's limited battery heavily depletes with added computation or hardware. Encryption through secure handshake protocols is the most secure option that we have presented but the added computational requirements may deplete a UAV's battery life rapidly. On the other hand Password/Token Authentication is the most convenient due to its scalability and is the second most secure option. Due to its combined benefits, we propose that Passphrase/Token Authentication is the best option for securing MAVLink.

VII. Acknowledgement

This work is part of an undergraduate senior design project conducted by students from the Stevens Institute of Technology Computer Science Department. We extend our appreciation to our project advisors, faculty members and the technical and administrative staff for their insightful guidance, mentorship, and encouragement, which were crucial to the success of this project.

References

[1] Z. Wang, Y. Li, S. Wu, Y. Zhou, L. Yang, Y. Xu, T. Zhang, and Q. Pan, "A survey on cybersecurity attacks and defenses for unmanned aerial systems," *Journal of Systems Architecture*, vol. 138, p. 102870, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762123000498

[2] N. H. N. S. O. C. A. Yaacoub, Jean-Paul A, "Robotics cyber security: vulnerabilities, attacks, countermeasures, and recommendations," *International Journal of Information Security*, 2021. [Online]. Available: https://doi.org/10.1007/s10207-021-00545-8

[3] A. Hristozov, E. Dietz, E. Matson, J. C. Gallagher, and M. Rogers, "Secure robotic vehicles: Vulnerabilities and mitigation strategies," in *2022 IEEE International Symposium on Technologies for Homeland Security (HST)*, 2022, pp. 1–6.

[4] L. CHAARI, S. CHAHBANI, and J. REZGUI, "Mav-dtls toward security enhancement of the uav-gcs communication," in *2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*, 2020, pp. 1–5.
[5] N. Sabuwala and R. D. Daruwala, "Securing unmanned aerial vehicles by encrypting mavlink protocol," in *2022 IEEE Bombay Section Signature Conference (IBSSC)*, 2022, pp. 1–6.
[6] N. A. Sabuwala and R. D. Daruwala, "An approach utilizing a random keystream generator to enhance the security of unmanned aerial vehicles," in *TENCON 2023 - 2023 IEEE Region 10 Conference (TENCON)*, 2023, pp. 1–5.
[7] N. Prapulla, S. Veena, and G. Srinivasalu, "Development of algorithms for mav security," in *2016 IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT)*, 2016, pp. 799–802.
[8] K. Kim and Y. Kang, "Drone security module for uav data encryption," in *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, 2020, pp. 1672–1674.
[9] I. F. Hashmi and E. Munir, "Securing the skies: Enhancing communication security for unmanned aerial vehicles," in *2023 17th International Conference on Open Source Systems and Technologies (ICOSST)*, 2023, pp. 1–6.
[10] J. Kim, Y. Choi, S. Jeon, J. Kang, and H. Cha, "Optrone: Maximizing performance and energy resources of drone batteries," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pp. 3931–3943, Nov 2020.
[11] S. R. Masadeh, S. Aljawarneh, N. Turab, and A. M. Abuerrub, "A comparison of data encryption algorithms with the proposed algorithm: Wireless security."
[12] E. Team. Heartbeat. [Online]. Available: https://networkencyclopedia.com/heartbeat/#Heartbeat-Security-Implications
[13] P. S. Tang, T. C. Lim, N. Nagappan, and L. Z. Wee, "Secure and encrypted heartbeat protocol," https://patents.google.com/patent/US20200162434A1/en, 2020, u.S. Patent Application No. 20/200,162,434.
[14] M. Z. Kamil, "Novel lightweight encryption api for iot device communication," https://scholarworks.calstate.edu/downloads/bv73c756d, 2023, accessed: 2023-09-24.
[15] "Ascon encryption scheme," https://ascon.iaik.tugraz.at/, 2023, accessed: 2023-09-24.
[16] J. R. R. R. Malathi R., "An integrated approach of physical biometric authentication system," https://www.sciencedirect.com/science/article/pii/S1877050916306214, 2016, accessed: 2024-02-25.
[17] A. Sarkar and B. K. Singh, "A review on performance,security and various biometric template protection schemes for biometric authentication systems," https://link.springer.com/article/10.1007/s11042-020-09197-7#Sec25, 2020, accessed: 2024-02-25.
[18] D. Hilm and D. Rahim, "Two-factor authentication and digital signing for an enterprise system utilizing yubikey," 2019. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:1335587/FULLTEXT01.pdf
[19] Pymavlink installation. ArduPilot Project. [Online]. Available: https://www.ardusub.com/developers/pymavlink.html
[20] A. E. Aaron Porter. How to start a mission using pymavlink. Youtube. [Online]. Available: https://www.youtube.com/watch?v=pAAN055XCxA&ab_channel=AscendEngineering