Securing the MAVLink Communication Protocol for Unmanned Aircraft Systems

Technical Report #CSSE14-02

Neil Butcher
Appalachian State University,
Boone, NC 28607
Email: butcherna@appstate.edu

Angela Stewart
Auburn University
Auburn, AL 36849
Email: aes0030@auburn.edu

Dr. Saad Biaz
Auburn University
Auburn, AL 36849
Email: biazsaa@auburn.edu

***Abstract*—Unmanned aircraft systems are gaining popularity in commercial and government applications. This, however, makes these systems susceptible to security attacks. This paper overviews some of the security considerations in unmanned aircraft systems and details an encryption protocol implementation. Because encryption can be very expensive in computational resources and battery consumption, an efficient and secure mechanism is developed that minimizes overhead using RC5 encryption on the existing communication protocol.**

I. Introduction

In an unmanned aircraft system (UAS) there are many constraints, such as power, time, and processor resources that complicate the addition of a security protocol into an existing system. However, attention must be paid to the most significant threats as it is essential to designing a security system to prevent attacks. The main security breaches that a UAS is subject to are packet forwarding attacks, eavesdropping, and hijacking. Such attacks render a UAS unable to perform to expectation. Many security system implementations exist to combat these attacks, however, upon examination, many implementation options were rejected because they were not optimal for resource constrained systems. In order to securely and efficiently transmit information across a channel an encryption cipher must be employed. Encryption is implemented on MAVLink, a communication protocol used in many unmanned aircraft systems, as well as the UAS used in this research.

A. Packet Forwarding Attack

One of the main categories of wireless network security intrusions are packet forwarding attacks. A packet forwarding attack is a common attack that involves data packets being delivered in an intentionally irregular way, such as dropping, duplicating, or modifying the packets to yield a compromised network[1]. For example, a UAS can be subject to denial of service when junk packets are added to the network. This causes the UAS to become useless because it fails to decipher useful commands from junk packets injected by an adversary. Additionally, packet forwarding attacks expose a UAS because crucial data can be modified to send false information. A packet forwarding attack can even drop packets entirely, thus preventing the base station or the planes from receiving crucial updates.

A specific example of a packet forwarding attack is a replay attack in which an adversary observes the transmission of a packet, then stores and retransmits the packet when it is advantageous to manipulate the UAS[2]. This type of attack can be particularly effective because data packets are often sent with passwords or encryption keys. If the intruder replays a packet with an encryption key, then it can gain access to the system by causing real nodes to believe that it is legitimate too.

Packet forwarding attacks can be very difficult to prevent because encrypted data can be replayed to gain access to the system. Therefore, encryption must guarantee semantic security–the property that plaintext will not be encrypted to the same ciphertext in subsequent encryptions[3]. If the plaintext is not encrypted to the same ciphertext, a replay of an old message with a previously held encryption key will be invalid. A common mechanism to guarantee semantic security is to implement encryption and exclusive or-ing (XOR) the encrypted plaintext with an encryption of random bits or the number of messages sent.[3]. The number of messages sent and received is included in the MAVLink header. This ensures that both receiver and sender are aware of the total number of messages transmitted, since packet loss is to be expected.

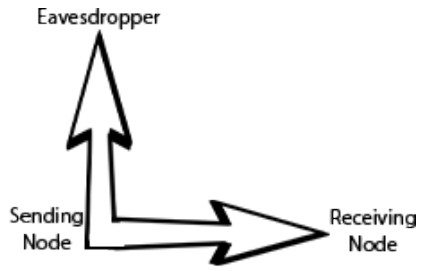

Figure 1. In an eavesdropping attack, the eavesdropper listens to communications from the sender to the receiver.

B. Eavesdropping Attack

The eavesdropping attack (Figure 1) is another common attack that must be accounted for in a security protocol designed for unmanned aircraft systems. In an eavesdropping attack an adversary silently listens to the communications between the base station and the nodes[4]. Often this type of attack is used to gain information about the system to develop a more advanced attack. A common solution to this problem is to encrypt data sent and received with a shared secret key that has been disclosed between the sender and receiver in a confidential manner. While an intruder might know about the presence of messages being sent, the intruder does not know the content of the messages when encrypted as described.

C. Hijacking Attack

One of the most serious forms of attack is hijacking (Figure 2). This attack is successful when the adversary gains control of the plane through mimicking the role of the base station[4]. A plane can be hijacked through a man-in-the-middle attack, an attack where the adversary is controlling the communication between the base station and a node. This gives the adversary full control of the UAS, while allowing the base station to believe that it is still in control of the plane[4]. This sophisticated attack can be prevented by using an uncompromised encryption key and thus preventing an intruder from taking control of a UAS.

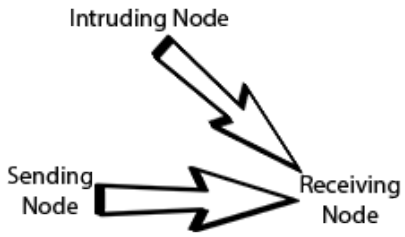

Figure 2. In a hijacking attack, the intruding node attempts to send commands to nodes in the system.

D. Rejected Security Protocols

There are multiple protocols that can be used to secure a wireless network. However, many were rejected because they were not an appropriate implementation for this research. For example, secure ad hoc routing calls for any message to be signed with a cryptographic authentication primitive. The receiver of the message verifies that the message was sent from a node with a legitimate authentication primitive[1]. This approach is problematic, however, because an intruder can gain control of an authenticated node and inject malicious messages into the network.

Localized detection (Figure 3) is a packet forwarding security protocol that does not provide the functionality needed for a UAS. Localized detection relies on the property of two nodes in a network being able to ”hear” each other’s transmissions. When the A node sends a packet to B, it knows the ultimate destination of the packet. If A does not hear B send the packet within a certain threshold of time, A marks B as being a potentially malicious node[1]. This approach is ineffective for this research project because packet loss is common amongst legitimate nodes. Even if a UAS fails to hear the proper transmission of a packet from the ground station, it does not mean that the ground station is a malicious node. This would only succeed in marking nodes that the UAS should be listening to as malicious, thus causing it to ignore appropriate commands. Additionally, this approach does not preclude a malicious node from modifying packets and properly forwarding them. The localized detection approach would not allow the UAS to detect such an intrusion and act on it.

A third rejected security protocol implementation is ACK-based detection (Figure 4). This method expects an acknowledgement (ACK) from a node when the node receives a message. Once a certain threshold of missed ACKs is reached, an error is triggered and the base node investigates the problem[1]. This approach is problematic for this research, again, because high packet loss is to be expected. A node that fails to receive packets is not necessarily a malicious one. Furthermore, because the base node does not investigate suspicious behavior until a threshold of warnings is reached, a malicious node can possibly do damage up to that point. This is an unacceptable risk for this unmanned aircraft system.

E. Solution: RC5

The security challenges described above can be remedied through the implementation of an encryption algorithm. This research makes use of RC5. RC5 is the encryption mechanism developed by Ronald Rivest in 1994. RC5 features data dependent rotates, where the rotation of values depends on the data[3]. The algorithm, known for its computational efficiency, when using 64-bit key size with 12 rounds, is susceptible to differential cryptanalysis with 2^{44} chosen plaintexts to reveal the corresponding ciphertext and compromise the entire

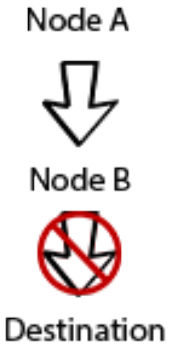

Figure 3. In localized detection, if node A does not hear node B properly send a packet to its destination, Node A marks B as being potentially malicious.

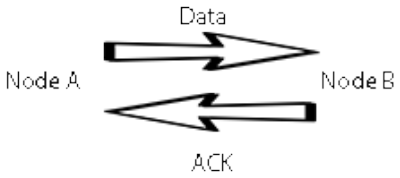

Figure 4. In ACK-based detection, the receiving node sends an acknowledgement back to the sender when it gets the data.

encryption protocol.

RC5 features a very complex key setup, and a very simple encryption and decryption algorithm which is the reason for its low computational overhead. Even while having this low overhead, RC5 remains a secure cipher for the purpose of this research. RC5 is not the most secure encryption algorithm, but it has proven sufficient for securing various communication protocols, especially ones that runs for short periods. RC5 was implemented in this UAS because it consumes less computational resources than more secure encryption mechanisms.

F. MAVLink

Micro Air Vehicle Link (MAVLink) is the header only, message-marshalling library used as the communication protocol between the ground station and plane[4]. The main components of a MAVLink message are the header, system ID, message ID, and payload. The header is used to classify the message as a MAVLink packet. The system ID identifies the system sending the message while the message ID identifies the type of message being sent. For example, the most common message to send is the heartbeat (ID = 0) which is constantly sent to ensure that the plane and ground station are properly connected and communicating. The payload of the message is the content inside it. The payload can contain fields such as the vehicle type, flight mode, positioning data, or commands to execute. These messages are sent as data packets between the ground station and plane to properly fly the plane.

II. Problem Description

Based on the need to prevent the attacks detailed above, this research project aims to solve the problem described as follows:

> Design and implement a security protocol for an unmanned aircraft system that prevents breaches in the confidentiality and integrity of the UAS with minimal computation overhead.

The confidentiality of a UAS is violated when an intruder is able to access information about commands sent to the UAS from the ground station as well as location information and other data sent in the opposite direction. The integrity of a UAS is compromised when commands from an intruder are received and performed by the UAS[4]. By protecting the confidentiality and integrity of a UAS, eavesdropping and hijacking attacks are prevented.

There are some major considerations to be aware of when designing a security protocol. These considerations include the inherent complexity of the MAVLink communication protocol, the existence of packet loss in the UAS, and the need for efficient implementation in a resource constrained environment. It is crucial that the security protocol implementation does not interfere with the performance of the system and cause further problems to autonomous flight.

This research attempts to implement a secure transmission of MAVLink messages into an existing UAS. Considering the format of MAVLink messages, this is a fairly complex problem because upon receipt of a message, the UAS initially checks that the message has a MAVLink header, as well as if it is the target of the message[4]. If the message is either not intended for this vehicle, or does not have a MAVLink header, it is disregarded. This makes header encryption impossible and message target encryption expensive. Encrypting the intended recipient has a large computational overhead because every message received with the MAVLink header has to be decrypted by the UAS before being able to discern if it was intended to receive the message. While this research uses a 16MHz processor that can perform encryption and decryption fairly quickly, the aforementioned extremely high overhead is unacceptable for an already resource constrained system. Therefore, this research allows the message recipient to be unencrypted and visible by potential intruders and accepts this minor breach in security as a reasonable compromise.

In addition to the need for an unencrypted MAVLink header and message targets, this implementation assumes that there will be considerable packet loss even without encryption. This fact makes it essential that transmission overhead is minimal. Increased length in packets makes reliable data packet transfer more difficult, thus rendering minimizing the length of encrypted packets essential. While packet loss is an inevitable part of this system it is imperative the loss of a packet is not due to the security system.

It is important to ensure that the instance of a single compromised plane will not compromise the entire system. For instance, if an adversary manages to gain access to the hardware of a plane, the secret key can easily be compromised. Because of this, an individual master secret key must be shared between the base station and each of the planes. This ensures that an adversary obtaining access to one key will not compromise the entire system. While this is effective at preventing the adversary obtaining control of all of the planes, the intruder could still relay false information back to the base station, appearing to be the compromised plane.

One final considerable factor in the proposed implementation of this research is power consumption. Power consumption will be increased by the extra workload of the processor as well as the transmission overheads of the longer packet length. The security protocol implementation aims to minimally reduce the amount of time the UAS is capable of staying airborne. Because telemetry is a power-intensive

operation, increasing the length of packets will significantly affect the performance of the UAS.

III. Literature Review

A significant amount of research has been done on unmanned aircraft systems and security protocols for similar networks. Researchers have looked into the overall vulnerability of MAVLink, as well as investigated security in resource constrained sensor networks and multilevel ad hoc networks. This previous research greatly impacted this research's implementation for securing the MAVLink communication protocol.

A. MAVLink Vulnerability Analysis

The MAVLink protocol was extensively tested for security shortcomings and critiqued during the research of Joseph Marty[4]. This research tests for a variety of security failures and proposes cryptographic implementations for securing the MAVLink protocol. Marty also performs benchmarking standards to quantify the cost of these solutions.

MAVLink, the message-marshalling library that serves as the communication protocol between the ground station and plane, identifies messages based on the header format[4]. The implication of this is crucial to the development of a security system. The header cannot be encrypted because messages are identified based on header format. If the header were encrypted, a MAVLink enabled UAS would not be able to distinguish messages being sent to it from noise. Since the message header cannot be encrypted, potential adversaries will always know when a MAVLink enabled vehicle is present. However, encryption of the message itself prevents an intruder from knowing the content of the message.

MAVProxy is a command line, Python based ground station that can command a MAVLink enabled UAS[4]. Marty's research made use of MAVProxy to connect to a UAS and maliciously eavesdrop on UAS data and commands and hijack the UAS. MAVProxy was also used to implement a denial of service attack on a UAS; however, denial-of-service attacks are out of the scope of this research. These attacks, implemented via a Python script, were used to exploit the complete lack of security protocols in MAVLink.

Marty's tests of the vulnerability of the MAVLink protocol revealed additional considerations for the design of a UAS security system. Power consumption while under attack and not under attack was measured. The tests revealed that attacking the confidentiality, integrity, or availability of the UAS did not have any significant impact on power consumption. However, as stated previously, when an encryption mechanism is in place, the implementation must not be so strenuous as to deplete battery life.

In addition to testing power consumption under various conditions, Marty also measured network latency. The amount of time from the initial issuing of the command to the time the command is performed was measured to observe the impact of security attacks. The study found no significant differences between the network latency of no attack, an eavesdropping attack, and a hijacking attack. However, the denial-of-service attack, implemented by issuing a "loiter" command on an infinite loop once per second, resulted in a non-negligible network latency[4]. Marty hypothesized that this network latency in the attack on availability results from an interrupt effectively being issued every second when the "loiter" command is repeatedly issued, thus delaying task processing.

While protecting against attacks on the plane's availability is out of the scope of this research, the presence of extreme network latency under said attacks reinforces the need for an efficient encryption algorithm. The security protocol put in place to protect against attacks on confidentiality and integrity must be computationally efficient enough to avoid a significant increase in network latency. Slowing down the response time to commands is harmful to a UAS because the plane would fail to respond to important commands to avoid obstacles in a timely manner. High network latency would also cause data sent to the ground station from the plane to be received later and potentially be irrelevant by the time the ground station receives it.

Previous research by Joseph Marty on the vulnerability of the MAVLink protocol revealed the need for a security system. Currently, no securing mechanism exists to prevent attackers from eavesdropping on the UAS, hijacking the system, or completely denying service of the UAS, rendering the whole system useless. The research conducted also pointed out the need for a low overhead security system that minimizes battery consumption.

B. SPINS: Security Protocols for Sensor Networks

Additional relevant research to the development of a UAS security protocol is the "SPINS: Security Protocols for Sensor Networks," research performed at Berkeley University by Perrig et al. Their research attempted to secure communication between sensor networks, a network of microcontrollers that record information about the environment and transmit those observations to a base station[3]. This differs in some manner from UAS information transmission because the information in a UAS is time sensitive. A sensor network and UAS are similar, however, because both systems are heavily resource constrained.

The SPINS research proposes SNEP, the Secure Network Encryption Protocol, that adds only an 8 byte transmission overhead[3]. SNEP differs from this research's implementation because the SPINS implementation assumes that very few packets are dropped. Because the SPINS implementation assumes little packet loss, the number of transmissions is used to guarantee semantic security. The encryption of the number of messages sent and the encrypted data is XORed to guarantee semantic security without adding a transmission overhead[3].This provides replay protection by ensuring the messages are delivered in the correct order. However, in this UAS research, because of packet loss, misalignment of the counters would occur unless the counter is included in every packet.

Perrig et al propose an authentication broadcast protocol, μTesla that uses a one-way hash and random initial key generation to authenticate messages broadcasted throughout the network[3]. Messages are encrypted using a hash function that computes output based on input, but the input cannot be computed based on the output. The random encryption key is initially created and broadcast to the network. Keys are then disclosed after a time delay. This method is unnecessary for our implementation because this research assumes that secure communication exists between the nodes and the ground station prior to launch. Therefore, any encryption keys need not be broadcast, because they can be safely shared prior to launch.

The research of Perrig et al on sensor networks provides insight into encryption algorithms optimal for resource constrained environments. Perrig et al chose the RC5 algorithm because it features fast encryption and decryption, low memory overhead, and cryptographic strength[3]. In the case of a UAS or sensor network protocol, RC5 works better as an encryption algorithm than RSA because RSA has a substantial memory overhead and is computationally inefficient. Since both unmanned aircraft systems and sensor networks are already fairly resource constrained the best implementation allows for the tradeoff between a reasonably secure implementation and a fast computation speed.

The sensor networks research provides a basis for the implementation proposed in this UAS security protocol research. There are many differences between the assumptions made in this research and the SPINS article, like time sensitivity and secure sharing of cryptographic keys rather than broadcast. However, the common assumption of a resource constrained system allows this UAS research to develop an implementation modified from the SPINS research. The SNEP protocol defined in the SPINS paper provides a basis for the development of the implementation used in this research[3].

C. Adaptive Security

The prevalence of UAS use in military applications has led to research on the optimal security protocol for multilevel ad hoc networks. Kong et al bring up security concerns in the military applications of these networks. These concerns include privacy, integrity, nonrepudiation, authentication, and availability[5]. Privacy (confidentiality), integrity, and availability are all discussed by Joseph Marty in his research on MAVLink vulnerabilities[4]. These terms take on the same meaning here. Additionally, in this context, nonrepudiation is used as the service of ensuring that authentication is done with a high level of confidence. Authentication refers to the validation of a piece of data and its sender as being legitimate. These five ideas form the basis of the research of Kong et al into securing multilevel ad hoc networks.

Kong et al investigated the UAV-MBN network, the unmanned aerial vehicle-mobile backbone network. This network consists of three levels of nodes with differing capabilities and functions. The regular ground nodes, typically soldiers in a military application, have limited computational power and limited-range broadcast. The second level consists of MBN nodes, the mobile backbone made of ground vehicles with more powerful computation capabilities, longer broadcast range, and more bandwidth. The third level of the UAV-MBN network consists of UAVs with "line-of-sight connectivity" with the MBN[5].

Due to the current multilevel structure of the UAV-MBN ad hoc network, there is risk of exposure due to a single point of failure. This can cause the entire system to be deemed corrupt. The third level of the network, the UAV, provides the authentication services for its single area theater. This single area theater consists of the UAV and its regular and mobile ground nodes. Any single area theater communicates with other theaters through the UAV. If the UAV is compromised, then the entire theater is compromised.

To combat the single point of failure downfall described above, Kong et al propose two modes for the UAV-MBN to operate in-infrastructure and infrastructureless mode. Infrastructure mode occurs when the third level of the multilevel ad hoc network is present[5]. That is to say, the network operates in infrastructure mode when the UAVs are present and not corrupted.

In infrastructure mode, the UAV acts as the certification authority, enforcing either implicit or explicit countercertification. A security certificate can be countercertified implicitly by enforcing frequent renewals. If a valid node becomes malicious, the frequent certificate renewals cause the newly malicious node to have an expired certificate that cannot be renewed, thus excluding it from the network. An alternative to implicit countercertification is explicit countercertification. A certificate is explicitly revoked when the certification authority, the UAV in this case, signs a countercertificate for a malicious node and adds that node to a certificate revocation list. Because only the UAV can sign the countercertificate, only the UAV can add nodes to the certificate revocation list[5].

When an intrusion is detected or the UAV is not available, the UAV-MBN network transitions to infrastructureless mode. Infrastructureless mode consists of only the first two levels of the UAV-MBN network, the regular and mobile backbone ground nodes[5]. In this mode, the certification authority is distributed, rather than central. Each ground node shares part of certification and countercertification and must use its own signature to sign certificates. A certificate or countercertificate is only valid if all ground nodes sign it[5].

The dual mode infrastructure-infrastructureless security system implementation on UAV-MBN networks is not an appropriate implementation for this research into a security protocol for MAVLink enabled unmanned aircraft systems. However, the research of Kong et al reinforces the need for a UAS to not have a single point of failure. The entire system cannot be compromised because one node is compromised. This is ensured by giving each plane two unique secret keys (one for each direction of encryption), shared safely through a USB prior to launch. This enforces security, even when there are compromised nodes. Additionally, the UAV-MBN research stresses the importance of authentication. Future work on the

MAVLink security protocol can take this into consideration to improve the security of the system.

IV. Implementation

This implementation of an encryption protocol makes use of a modified AVR Crypto Lib to implement the RC5 algorithm. This library is used in the existing UAS code as the method of securing MAVLink.

A. Encryption and Decryption

In this implementation, encryption and decryption are done in 64-bit blocks. Library calls to the modified AVR Crypto Lib are made using these 64-bit blocks as parameters to encrypt and decrypt the data and commands sent between the ground station and plane. Currently the key is hardcoded into the ArduPilot.ino, xbeeIn.cpp and xbeeOut.cpp files. It is also important to note that the AVR Crypto Lib implementation of RC5 is limited to a 64-bit key size and a maximum of 255 rounds.

Commands, such as setting the next waypoint and stream rate changes, are sent from the ground station to the plane. Messages with such commands are encrypted at the ground station by encrypting the payload in 64-bit blocks. The checksum is then computed, to avoid issue with the plane receiving messages. The encrypted message is then sent to the plane to be decrypted. Prior to decryption, the plane looks at the checksum again. This forces the checksum to be computed twice, which adds an additional overhead. Once the message is received, 64-bit blocks are decrypted to return the payload to plaintext.

Encryption is done from the plane in the same manner as encryption done in the opposite direction. The payload is encrypted in 64-bit blocks and the checksum is computed after encryption to ensure that the message is properly received by the ground station. Once the plane sends encrypted messages back to the ground station, information is decrypted by first checking the message ID. If statements determine the type of message that is being received. Relevant fields to the message type are assigned to a space in a 64-bit array, made of two 32-bit indices. Each of these 64-bit blocks is then decrypted. This encryption and decryption code can be examined in the appendix.

B. Implementation Challenges

Several characteristics of MAVLink and the UAS caused challenges for implementation. The MAVLink checksum as well as field reordering had to be accounted for in implementation. By using an existing communication protocol, these challenges were exacerbated, in contrast to if they had been accounted for when initially designing MAVLink. In essence MAVLink was not developed with security needs in mind, and this greatly extended the overheads of securing the system, such as the necessity of recomputing the checksum. Additionally, issues with the XBee radios used in this UAS caused problems for testing the secured protocol.

MAVLink makes use of a checksum to verify that data is not altered from the creation of a message to its receipt. The format of a message is checked to make the existence of corrupted message fields less likely. This is done by adding one byte to the checksum that is based on the length of the message, as well as message field types and names[6]. This check is particularly useful when a plane and ground station are using slightly different versions of the MAVLink code. A message must have the proper characteristics outlined above, regardless of the version of the code, to make messages with the same length but different fields be rejected.

Additionally, MAVLink uses the checksum to determine if a message was changed. MAVLink takes the checksum of a message and stores it in an array. This is added to the end of the message[6]. If the message itself, rather than its format, changes, then the checksum value previously calculated will be incorrect and the message will be rejected. This happens in addition to checking the format of the message, as described above.

The checksum presents problems for an encryption based security protocol because encryption changes the content of messages. The plaintext is changed to ciphertext and the checksum will detect that as a change in the message. A solution to this would be to perform encryption before calculating the checksum. However, this is difficult because the implementation would have to ensure that decryption is performed after the checksum is looked at again. However, in MAVLink, the checksum is created and checked in the XBee firmware, which cannot feasibly be modified in this research. The other alternative is to encrypt after the checksum is calculated and decrypt before it is checked again. This, however, poses the same problem with the XBee firmware.

Prior to calculating the checksum, MAVLink reorders the message fields according to size[6]. This causes further difficulties in an encryption based security protocol. If the fields are reordered after encryption, the decryption might not return to the proper plaintext because the order is different. Any security protocol implementation on MAVLink must take the checksum and field reordering into account in order to properly transmit messages.

While the checksum and field reordering were an issue for implementation, the radios used in this research proved to be a problem in testing. This UAS uses the XBee, made by Digi International, to communicate between the ground station and plane. This research aimed to duplicate the methods used in previous research exposing the security vulnerabilities of MAVLink[4]. However, multiple XBees were not able to be configured to send information to the plane. That is to say, this research was not able to implement an uncompromised ground station and a malicious intruder to both communicate with the plane. Further research might prove this possible with the system configuration used in this research.

V. Results

With little field testing and analysis it can be concluded that encryption and decryption is performed in both directions of

communication. This shows that the MAVLink protocol can be secured through encryption and decryption although the computational efficiency in a real time UAS is still to be determined. The large computational overhead of encryption and decryption can be a factor in preventing the UAS from working properly. Field testing will determine how significantly the encryption affects the performance of the UAS.

Overall time constraints limited the development of results in our research, but this research did successfully implement a low cost encryption mechanism to the UAS. The implementation allows what is widely regarded as an easy cryptographic break, but under the assumption of a short flight time the key is not feasibly breakable. This ensures security of the system, and, with a proper key transmission, a unique key would be generated for each direction of communication. This ensures a more difficult cryptographic break with a minimal overhead.

This research provides a strong start to securing the MAVLink protocol. While there is still much more to be done, this implementation provides a good foundation of security by encrypting the communication in the UAS. This prevents adversaries from easily obtaining information about the system unless they break a cryptographic key. While much work is still to be done on this project, significant progress has been made.

VI. Future Work

This implementation of encryption leaves future work to be done to protect MAVLink against vulnerabilities. Accounting for these vulnerabilities will improve the security of the system and protect against more sophisticated attacks on the communication protocol. Furthermore, extensive testing on the encryption protocol must be done to expose any further vulnerabilities of the UAS and analyze its performance with the secured MAVLink protocol.

A. Unencrypted Messages

The security protocol implemented in MAVLink currently will accept any MAVLink message and attempt to decrypt it. For example, the plane will try to receive unencrypted messages and decrypt them according to the algorithm used. This is problematic because decrypting plaintext could still yield valid commands. For instance, if an unencrypted message that was not sent from the ground station has a new waypoint for the plane, it is very likely that the value of the waypoint will still be decrypted to some valid coordinates on earth. This would cause the plane to change its course to that new waypoint, no matter where that location is. Other plaintext commands sent to the plane are less likely to yield valid values, like stream rate changes, because there are limited possible values in those sets, but fields like the next waypoint are very much vulnerable.

Future work can attempt to secure this vulnerability in a number of ways. An improved implementation could narrow the possible values for fields like the next waypoint. That is to say, a future implementation could only accept waypoint values that are within a certain radius of the plane's current position. That would improve the security of the plane by making it less likely that an unencrypted command would be decrypted to a valid value, however, it is still not impossible. This approach would have to be used for all fields that have a wide range of possible values.

Another approach to protecting against the unencrypted message vulnerability would be to check for encryption prior to attempting to decrypt messages. This is tricky, however, because there currently appears to be no defining characteristic of encrypted messages that can be checked for prior to decryption. If a method to check for encryption prior to decryption arises, then the unencrypted message vulnerability can be eliminated much more thoroughly than through previously discussed methods.

A final option to protecting against unencrypted messages is implementing an authentication protocol. An authentication protocol would use a digital signature to verify that the message is being received from a legitimate source. This potential implementation is especially difficult in practice because authentication would cause significant burden on the resource constrained system. As stated previously, processing power is limited and commands must be executed in a timely manner. Authenticating the sender of every message would slow down the UAS, however, it would provide an increased level of security for any message sent from an illegitimate sender.

B. Counter Mode Encryption

This implementation of an encryption protocol for MAVLink does not provide semantic security or prevention against replay attacks. Counter mode encryption XORs ciphertext with the encryption of the number of messages sent. This added layer guarantees that messages will not be replayed later by an adversary because some plaintext does not become encrypted to the same ciphertext. In other words, implementing counter mode encryption would make the MAVLink security protocol semantically secure.

C. Secure Key Transmission

Currently, this implementation of a security protocol for MAVLink includes a hard coded security key. It is important that the keys be generated randomly so that an adversary cannot determine the keys generated. Future work can improve this by securely sharing keys between the nodes and the base station prior to launch. The keys should be distributed only when the plane is connected to the ground station via USB. This would provide increased security by ensuring that keys are safely shared prior to launch. It is important to note that unique keys for each direction of communication should be generated.

D. Testing

Extensive testing is still needed on this security protocol implementation. Methods of testing the security protocol implementation would include benchmarking mission completion time, network latency, battery usage, and processor usage with

and without the encryption implementation. A mission should be generated on the unsecured MAVLink protocol and the above fields measured when that mission runs. This can then be compared to the same mission ran on the secured MAVLink protocol.

The possibility exists that the secured protocol can cause the mission to not be completed in a reasonable amount of time. Furthermore, network latency could be so high with the security protocol that important commands are not processed in a timely manner or data becomes irrelevant. In either of these cases, the encryption implementation will need to be improved for efficiency.

It is also possible that battery or processor usage would be too high to feasible use this encryption implementation. Increasing battery usage by a large amount would result in decreased maximum mission time. The plane would not be able to stay in the air long enough to do any useful work. Furthermore, an overly busy processor could result in the slow processing of commands and data described above. The overused processor problem can be solved by upgrading to a superior processor; however, it can also be solved in software by increasing encryption efficiency. Battery usage is less easily solved, but could be improved by a more efficient encryption algorithm that uses fewer resources.

Finally, the security protocol in MAVLink should be tested when the system is under attack. Comparison between the behavior of the unsecured and secured MAVLink can measure the success of the implemented encryption mechanism. Methods similar to those used in the previously described malicious MAVProxy script[4] can be used to attack the secured MAVLink protocol. An eavesdropping or hijacking attack can be implemented to test the response of the secured UAS. To more thoroughly test the system, the previously discussed mission completion time, network latency, battery usage, and processor usage should be measured when the secured protocol is under attack and when it is not.

VII. Conclusion

The vulnerable MAVLink protocol can be secured by an encryption mechanism implemented in the existing code. This problem on securing an existing communication protocol has shown that security must be accounted for in the original design of the system, otherwise incompatibility issues, like with the checksum, will cause problems when attempting to secure the system later. Furthermore, security is essential to any communication protocol to ensure safe and reliable message transmission across a network. Further work should be done to test the encryption implementation and rid the system of its current vulnerabilities through secure key generation and counter mode encryption. However, this research has revealed that MAVLink can indeed be secured to make the entire communication protocol more reliable.

Appendix A
From Ground Station to Plane

A. Encryption in Ground Station

```
// This code was added in xbeeOut.cpp
  mavlink_message_t mavlinkMsg;
  uint8_t chan = MAVLINK_COMM_0;
  // Refer to message_definitions for parameter
      explination
  switch (cmd.commandID) {
    case COMMAND_NORMAL_WP: //waypoint command
    case COMMAND_AVOID_WP: //waypoint command
      mavlink_msg_mission_item_pack(sysid, compid, &
          mavlinkMsg,
            sysid, serial_compid, WPSendSeqNum,
            MAV_FRAME_GLOBAL, MAV_CMD_NAV_WAYPOINT,
            2, 0, 20.0, 100.0, 1.0, 0.0,
            cmd.latitude, cmd.longitude, cmd.
                altitude);

      for(int x = 0; x < (mavlinkMsg.len / 8); x++)
      {
        rc5_enc((void *) (mavlinkMsg.payload64 + x),
            ctx); // enc message
      }
      // recompute the checksum
      mavlink_finalize_message(&mavlinkMsg,
          mavlinkMsg.sysid, mavlinkMsg.compid,
          mavlinkMsg.len, 254);


      break;
    case COMMAND_SET_ID: //change id commmand
      mavlink_msg_param_set_pack(serial_compid,
          compid, &mavlinkMsg, sysid, compid,
             "SYSID_THISMAV", (unsigned)cmd.param,
                 MAVLINK_TYPE_UINT64_T);
      for(int x = 0; x < (mavlinkMsg.len / 8); x++)
      {
        rc5_enc((void *) (mavlinkMsg.payload64 + x),
            ctx); // encrypt
      }
      // recompute the checksum
      mavlink_finalize_message(&mavlinkMsg,
          mavlinkMsg.sysid, mavlinkMsg.compid, 23,
          168);
      break;
    case COMMAND_SET_STREAMRATE: //change stream
        rate command
      mavlink_msg_param_set_pack(serial_compid,
          compid, &mavlinkMsg, sysid, compid,
            "SR0_POSITION", (unsigned)cmd.param,
                MAVLINK_TYPE_UINT64_T);
      for(int x = 0; x < (mavlinkMsg.len / 8); x++)
      {
        rc5_enc((void *) (mavlinkMsg.payload64 + x),
            ctx);
      }
      // recompute the checksum
      mavlink_finalize_message(&mavlinkMsg,
          mavlinkMsg.sysid, mavlinkMsg.compid, 23,
          168);
      break;
    default:
      break;
  }
```

B. Decryption in Plane

```
// This is the decryption in the GCS_Mavlink.ino
    file
for(int x = 0; x < (msg.len / 8); x++) // go through
     each block
{
    rc5_dec((void *) (msg.payload64 + x), ctx); //
        decrypt it
}
handleMessage(&msg);
```

APPENDIX B
FROM THE PLANE TO THE GROUND STATION

A. Encryption in Plane

```
 // In GCS_MAVLink.ino
 // This is where data is encrypted and then sent.
 static void NOINLINE send_au_uav(mavlink_channel_t
     chan)
{

  uint32_t latlng[2];
  uint32_t altlat[2];
  uint32_t lngalt[2];
  uint32_t spd[2];
  uint32_t dist[2];
  latlng[0] = current_loc.lat;
  latlng[1] = current_loc.lng;
  altlat[0] = g_gps->altitude;
  altlat[1] = next_WP.lat;
  lngalt[0] = next_WP.lng;
  lngalt[1] = next_WP.alt;
  spd[0] = g_gps->ground_speed;
  spd[0] = g_gps->ground_speed;
  dist[0] = ahrs.yaw_sensor;
  dist[1] = wp_distance;
  rc5_enc((void *) (latlng), ctx);
  rc5_enc((void *) (altlat), ctx);
  rc5_enc((void *) (lngalt), ctx);
  rc5_enc((void *) (spd), ctx);
  rc5_enc((void *) (dist), ctx);

    mavlink_msg_au_uav_send(
        chan,
        latlng[0],
        latlng[1],
        altlat[0],
        altlat[1],
        lngalt[0],
        lngalt[1],
        spd[0],
        spd[1],
        dist[0],
        dist[1],
        g.command_index); //DJ_5/24/13 - Changed
            heading to ahrs.yaw_senser to mimic
            send_location message
```

B. Decryption in Ground Station

```
        // This is where the data is received by the
             base station and then decrypted
        if(message.msgid == MAVLINK_MSG_ID_AU_UAV)
      {
        //ROS_INFO("Received AU_UAV message from
            serial with ID #%d (sys:%d|comp:%d):\n",
             message.msgid, message.sysid, message.
            compid);
        uint32_t latlng[2];
        uint32_t altlat[2];
        uint32_t lngalt[2];
        uint32_t spd[2];
        uint32_t dist[2];

        au_uav_ros::Telemetry tUpdate;
        mavlink_au_uav_t myMSG;
        mavlink_msg_au_uav_decode(&message, &myMSG);
        latlng[0] = myMSG.au_lat;
        latlng[1] = myMSG.au_lng;
        rc5_dec((void *) (latlng), ctx);
        altlat[0] = myMSG.au_alt;
        altlat[1] = myMSG.au_target_lat;
        rc5_dec((void *) (altlat), ctx);
        lngalt[0] = myMSG.au_target_lng;
        lngalt[1] = myMSG.au_target_alt;
        rc5_dec((void *) (lngalt), ctx);
        spd[0] = myMSG.au_ground_speed;
        spd[1] = myMSG.au_airspeed;
        rc5_dec((void *) (spd), ctx);
        dist[0] = myMSG.au_target_bearing;
        dist[1] = myMSG.au_distance;
        rc5_dec((void *) (dist), ctx);
        myMSG.au_lat = latlng[0];
        myMSG.au_lng = latlng[1];
        myMSG.au_alt = altlat[0];
        myMSG.au_target_lat = altlat[1];
        myMSG.au_target_lng = lngalt[0];
        myMSG.au_target_alt = lngalt[1];
        myMSG.au_ground_speed = spd[0];
        myMSG.au_airspeed = spd[1];
        myMSG.au_target_bearing = dist[0];
        myMSG.au_distance = dist[1];
        ROS_INFO("reLattitude: %d, Longitude: %d ",
            myMSG.au_lat, myMSG.au_lng);
        convertMavlinkTelemetryToROS(myMSG, tUpdate)
            ;
}
```

REFERENCES

[1] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang, "Security in mobile ad hoc networks: challenges and solutions," *Wireless Communications, IEEE*, vol. 11, no. 1, pp. 38–47, 2004.

[2] B. Maharrey, "Authentication protocols, their issues and our solutions."

[3] A. Perrig, R. Szewczyk, J. Tygar, V. Wen, and D. E. Culler, "Spins: Security protocols for sensor networks," *Wireless networks*, vol. 8, no. 5, pp. 521–534, 2002.

[4] J. A. Marty, "Vulnerability analysis of the mavlink protocol for command and control of unmanned aircraft," 2013.

[5] J. Kong, H. Luo, K. Xu, D. L. Gu, M. Gerla, and S. Lu, "Adaptive security for multilevel ad hoc networks," *Wireless Communications and Mobile Computing*, vol. 2, no. 5, pp. 533–547, 2002.

[6] Qgroundcontrol. [Online]. Available: http://www.qgroundcontrol.org/