Implementation of an Artificial Immune System to Mitigate Cybersecurity Threats in Unmanned Aerial Systems

Meagan Shivers
Department of Aerospace Engineering
Embry-Riddle Aeronautical University
Daytona Beach, USA
shiversm@my.erau.edu

Christian Llanes
Department of Aerospace Engineering
Embry-Riddle Aeronautical University
Daytona Beach, USA
llanesc@my.erau.edu

Maxwell Sherman
Department of Computer Science
Gonzaga University
Spokane, USA
msherman3@zagmail.gonzaga.edu

This project aims to use an artificial immune system to detect cyber-attacks on the dynamics of a minidrone. Using linear transfer functions, a zero-dynamics attack was designed to be injected into the system. To further explore how zero-dynamics translates from linear to non-linear systems, a similar attack was made using a random value rather than an unstable zero of the system. These attacks were then injected via the ground station computer. An offline artificial immune system was created and used to test the effectiveness of detecting the attacks. The immune system uses a discrete, binary, non-self recognition algorithm to create detectors and find disturbances in the data.

Keywords—artificial immune system, zero-dynamics, non-linear, attack, cybersecurity

I. Introduction

Unmanned aerial vehicle (UAV) warfare, commonly known as "drone warfare," has become central to armed conflicts in the 21st century. It enables countries to perform aerial surveillance and execute precise air strikes without the need for pilots to physically be present. In addition, a significant amount of research is dedicated to UAV autonomy, to allow the vehicles to recover from various malfunctions or attacks.

It is of paramount importance to all militaries with UAVs to ensure the security and reliability of their vehicles due to their significance. In December 2011, Iran captured an American-made Lockheed Martin RQ-170 Sentinel, which was used for a variety of surveillance and other data-collecting purposes. Iran commandeered the UAV, landed it, downloaded and decoded all of the information on it, and produced new vehicles based on their reverse-engineered design [1]. Ensuring the security of these UAVs means ensuring the security of the potentially critical data on them, as well as the technology within the vehicles themselves.

Desired security measures entail ensuring the UAV can detect intrusions, create countermeasures, execute the countermeasures, and recover in the event of a successful attack. If there is physical damage or faulty sensor data, for instance, the UAV must work to stay airborne, and enter a recovery sequence as determined by the operator (fly in circles, return home, self-destruct, or any combination of tasks like these).

II. Background and Motivation

Machine learning is a popular technique where computers, without explicit instructions, find patterns and derive solutions often much more efficiently and effectively than their human counterparts. One form of machine learning exists in the form of the genetic algorithm, where natural selection is simulated on a set of solutions. Potential solutions are evaluated based on a user-defined heuristic known as *fitness*. The weaker solutions are removed and the stronger solutions are paired, bred, and mutated [2]. This process repeats usually until a certain fitness threshold is reached, or until a certain number of generations has been produced. These sorts of algorithms help find robust solutions with less human intervention.

An artificial immune system (AIS) expands on the genetic algorithm. Modeled after a biological immune system, the AIS treats the varying problems it must solve as antigens, and solutions to these problems as antibodies. It uses a genetic algorithm to generate antibodies: sets of basic instructions that potentially solve the problem at hand. A pre-existing knowledge base is used to employ negative selection which prevents any known bad solutions from entering the gene pool. Previous solutions that worked well are stored in a database, comparable to biological memory cells, to quickly recall how to solve the problem again rather than needing to create a solution from scratch [2]. A training set, then, can act as a "vaccine."

AIS have proven useful in the past with UAVs. AIS have been successfully employed in UAVs to compensate for subsystem failures, low power resources, navigation, and even

monitor the health of components as they age [3]. While AIS are not suited for intrusion detection, they are very capable of compensating for attacks once they happen. This project's goal is to prove that an AIS can be used in a cybersecurity aspect to aid the control of a UAV that has been pervaded by external and unwanted persons.

III. Previous Works

In prior research, AIS have been utilized to create fault tolerant control systems on aircraft. Fault tolerant systems account for failures in sub-systems of aircrafts, such as the actuators and structures, and adjust the control laws of as needed. AIS have been integrated to these systems as the method of machine learning. It has been shown that artificial immune systems have high positive identification rates within these fault tolerant systems [3]. The AIS was able to identify what mode of failure occurred and adjust the control laws as necessary.

Within the artificial immune system, the use of a self-non-self-identification approach is used to recognize changes in patterns of data. These systems must be "trained" with nominal data to create a self-space to be used in the pattern recognition scheme. A self-space can be represented as a binary string that contains a map of the system's usual activity. Previous work has been done to take real-value sets of data and create a binary self-space from them. From this self-space, different algorithms for matching rules that use the negative selection concept were used to train and test the immunology system using this self-space [4]. It was found that the r-chunk method with a sub-string size of 10 was found to be the most effective at detecting these changes while maintaining a low false alarm rate.

The robustness of the AIS should allow it be effective for sensing even stealthy attacks. One form of stealth attacks know is the zero dynamics attack. When a non-linear, real system is linearized, a finger print containing the information about the system's dynamics is found and summarized in a control matrix. The matrix contains information about the system such as stability, damping, and frequency. The limit of control design and stability are noted by the zeros found within this linearization. Zero dynamics attacks exploit the unstable zeros found when the system is discretized at a specific sample time. An attack using the unstable zero as an input command is injected into the system and causes it to become unstable [5]. When viewed in continuous time, the unstable nature of the system is very apparent. Sensors, however, do not work in continuous time. If a sensor works at the same sample time as that of the attack, it cannot be seen and could go completely undetected by a monitoring system. The sample time conundrum makes zero dynamics attacks very difficult to detect. Adaptive control systems, such as an L1 multirate, have been implemented to negate these types of attacks in the past [6]. The controller did an excellent job at removing the attack from the system while maintaining stability.

This project aims to expand upon these works by creating an artificial immune system that uses sets of binary representation strings for the signals sent from the sensors to detect changes in the sensors that have been caused by cybersecurity breaches. The artificial immune system will combine the first two previously discussed works to create a novel approach to mitigate the effects of cyber-attacks. This approach allows for a database of solutions to be created that can be expanded upon as new technology is created and added to unmanned aerial systems for more accurate sensing. Zero dynamics attacks will then be used as the means of testing the effectiveness of the newly created adaptive control system.

IV. Methodology

A. Attack Design and Implementation

This study focuses on using zero-dynamics attacks to create unstable systems using stealthy attacks that are based off of the dynamics of the system. To create these attacks, the quadcopter was flown via commanded pitch and roll while the VICON system tracked and logged the movement of the aircraft. This real-time data was then used in MATLAB's System Identification Application to generate transfer functions that describe the systems motion in inputs and outputs. Two different types of transfer functions were generated: pitch command-to-x-position and roll command-to-y-position.

These transfer functions were then discretized to with sample times of 0.1, 0.03, and 0.05 seconds. After discretizing the transfer functions, their zeros were found. In zero dynamics attacks, unstable zeros are exploited by implementing them into the system at the same sample rate which the system is being discretized.

Linear system simulations of the transfer functions were then created in Simulink to develop the attacks. Fig. 1 shows the simulation environment used to do so. The attack constant varies based on the magnitude and sign of the unstable zero used to develop the attack. It is attack specific and is chosen based on how quickly the attack grows.

The gain value was used to turn the attack "on" or "off" based to view the responses of the linear system with or without the attack. The attack is added to the reference value and then fed into the transfer function. In the attacks generated, the reference is either a roll or pitch orientation. A reference value of 0.0001 was used in generation of all the attacks. A reference value of zero was preferred, but many

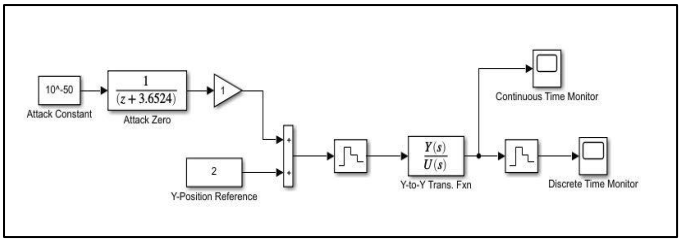

Fig. 1. Linear simulation environment used to develop zero-dynamics attacks.

times the attacks would occur far too rapidly and a small, near zero reference value solved this issue. To discretize the continuous-time transfer function, two zero order holds, one before and one after the transfer function, were placed into the simulation environment and had their sample times changed accordingly with the attack

As attacks were being created, it was noted that the magnitude of the unstable zero played an important role in the power of the attack constant. Larger magnitude zeros grew much faster and had to have much smaller constants to remain stealthy. Due to time constraints, an attack on the roll command with a sample time of 0.03 seconds was focused on. The attack value was -1.0423 and the attack constant was 10^{-5}.

To further study how zero dynamics attacks work in the non-linear system, a similar attack was designed with a random value rather than an unstable zero. The attack value used was -1.052 and the attack constant 2×10^{-7}. These values are very similar to that of the unstable zero attack. Fig. 2 shows the two attacks compared to one another as well as the differences in the attacks when created using the linear system.

It can be seen that when the linear system is sampled at 0.03 seconds, the unstable zero attack is completely undetected while the random value attack is seen almost immediately. Both attacks have similar magnitudes which is shown in the last two plots in the figure. These two attacks were used in simulation as well as in the real system to gather data for the immune system to process.

The quadcopter project in Simulink is a previously created simulation of the UAV used in this project. This non-linear model includes the flight controller, sensors, and a non-linear airframe model of the quadcopter. Its responses to the attacks gave a first glance at how the real system might respond. The attacks were added to reference command values, just like in the linear system, and data from the estimator and sensors were logged for analysis. Since the attack focused on was a roll command, data from the roll angle, roll rate, y-direction velocity component, and y-position were used to analyze the effects on the dynamics.

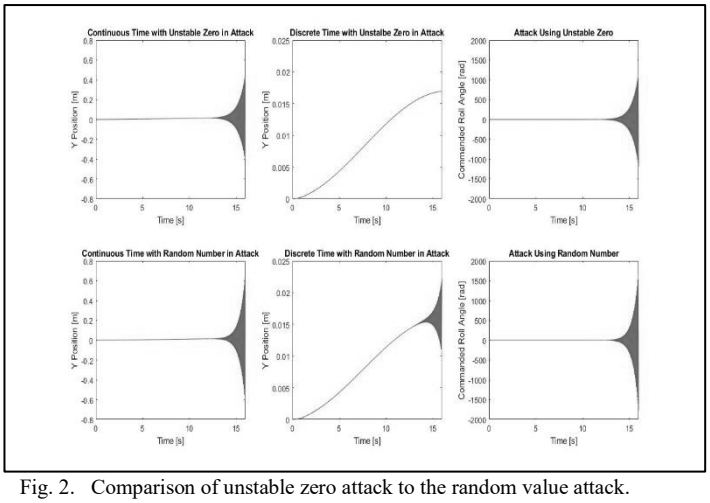

Fig. 2. Comparison of unstable zero attack to the random value attack.

To implement the attacks into the real system, the flight controller was edited to an orientation reference and the attacks added in at their respective sample time. The attacks began after five seconds of flight time to give the quadcopter time to stabilize. The VICON system was used to mock GPS and track the movement and orientation of the quadcopter while it flew. The data from the estimator, sensors, and VICON were all collected to analyze the real system's response to these sorts of attacks.

An RC controller was also connected to the quadcopter. Rather than reading a single reference command that was initially stated within the flight controller, the RC controller allows a pilot to control the quadcopter in real time. Eventually, the attacks will be sent via this communication means rather than having them preprogrammed into the flight controller. In real scenarios, an attacker would not have access to compile the flight controller, but could intercept or handle the radio frequency at which a pilot's controller is operating.

B. Hardware Setup

As stated earlier, due to time constraints, the focus of cyber-attacks on the drone was shifted to those where the attacker can gain enough control of the drone to inject a zero-dynamics attack. This means that denial-of-service attacks such as jamming or spamming corrupt packets, as well as false GPS coordinates/altitude sensor spoofing are no longer of interest.

The main objective is to verify the effectiveness of the AIS in real flight hardware. The Parrot Mambo minidrone was chosen to be used due to its robustness in handling crashes and relatively cheap cost. The Parrot Mambo also has the ability to be interfaced with Simulink using the Parrot Minidrones Support toolbox. Initial testing is done using the Aerospace Blockset Quadcopter example model from the *asbQuadcopterStart* MATLAB function because it provides an estimator and controller ready to fly. To determine which binary algorithm would be best for the AIS, this non-linear model was used to collect preliminary data to create and test the immunology.

The model was extended to support multiple controllers and a VICON motion capture camera system for local positioning. The Simulink model was designed to be user-friendly with buttons to select from the three different controllers before compiling and uploading to hardware. The three controllers are a simple cascaded PID controller, a Nonlinear Dynamic Inversion (NLDI) controller, and a L1 controller. The NLDI controller incorporates an Adaptive Neural Network (ANN) designed by a graduate student at the Embry-Riddle Aeronautical University Advanced Dynamics and Controls Lab (ADCL) [7]. This ANN will aid in extending the time to crash so that the AIS has enough time to detect the attack and respond.

The Parrot Mambo uses Bluetooth to communicate with the ground control station computer (GCS) running Simulink

and sending the VICON position data via UDP protocol. This is an unsafe protocol, but it was used for the sake of getting the project groundwork completed and making sure the other systems worked with a reliable transport layer for the high-bandwidth and real-time demands of drone localization. The original PID controller was used to collect nominal data to create the immune system.

V. RESULTS

Based upon many preliminary tests, it was determined that the optimal binary, negative selection algorithm for the AIS, in this application, was the hamming method. This algorithm was chosen due to its high identification rate, low false alarm rate, and low computation time. This method was used to train the immunology to detect both the attacks in the simulation and real system environment. The real system data was tested with two different data representations: four bits and six bits.

A. Simulation Results

Nominal data about the roll angle was collected by giving a series of roll and pitch commands to the simulation. The roll angle nominal data was then mapped to a binary self-space and used to create the immune system. A repertoire size of 1500 was set. The simulation's data generated 1433 detectors

To validate that the immune system was working properly, three test flights were flown in the simulation and tested against the immune system. These flights had varying roll angle commands given at various lengths of time. All of these tests had a low false alarm rate meaning that the immune system was functioning well.

After validation, both attacks were implemented into the roll command of the simulation. The simulation was ran for twenty seconds while data was logged. Self-spaces were then created from the data and tested against the immune system. Fig. 3and fig. 4 show the immune system's detection scheme.

In both of the attacks, the roll angle of the quadcopter grows unbounded until the quadcopter crashes. The immune system detects the attacks very well due the large growth and oscillations. In these figures, 1000 data points corresponds to five seconds of simulation time. The immune system, in both cases, begins to detect the attacks around 12.5 seconds and continues activate detectors until the quadcopter crashes. These results show that the AIS can detect attacks that will grow in an unbounded fashion.

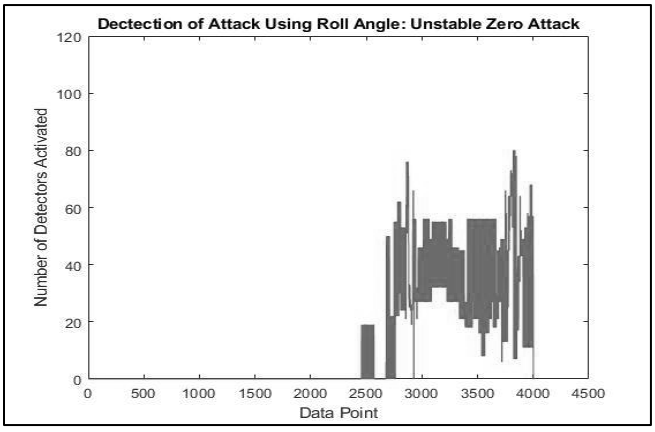

Fig. 3. Detection scheme for simulation under unstable zero attack.

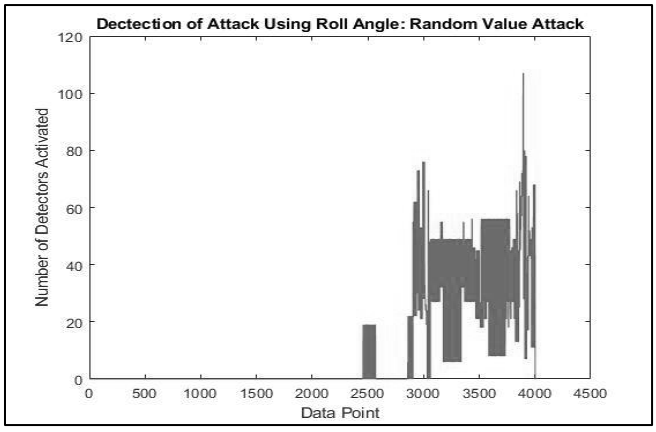

Fig. 4. Detection scheme for simulation under random value attack.

B. Real System Results

As in the simulation, five test flights were conducted to collect nominal data on the roll angle of the quadcopter. The commands were given and the roll angle response was logged. Two different immune systems were created with the data. The first immune system was created using a 4-bit representation of a time-slice in the self-space, a repertoire of 1500 in the hamming method, and a threshold value of 12. The second had a 6-bit representation of a time-slice in the self-space, a repertoire of 2500, and a threshold of 18. The number of time-slices per string was kept at a value of four for each of the immune systems.

The quadcopter was flown using the RF controller to collect data to validate the immune systems. The test flight had no pre-described path. Simple roll and pitch commands were given using the RF controller and the roll angle response logged. A binary-self space was created from the data and tested against the immune system. Both immune systems had a low false alarm rate and both immune systems were validated.

After validation, the unstable zero attack and the random value attack were implemented into the system six times each. The roll angle was logged for each of the attacks and used to create self-spaces to be tested against the immune system. In both attacks, the quadcopter crashes after fifteen second of flight time allowing the attacks ten seconds to grow. The data for the first five seconds was trimmed due to the quadcopter using this time to stabilize.

When comparing the two attacks, it is apparent that the unstable zero causes the quadcopter to roll at a higher magnitude than the random value attack. This trend is present in all of the other test flights as well. Using the dynamics of the quadcopter to create the attack causes a more severe

failure in the system. These responses, however, are much smaller than the simulation's response to the attacks. The lack of exponential growth could be due to saturations and limitations of the motors that were not properly modeled within the non-linear simulation.

After creating the self-spaces from the roll angle data, both immune systems were used to detect the attacks. The immune systems had similar computation times and were very fast in going through all of the data. Fig. 5 compares the detection scheme for both the immune systems using the 4-bit scheme while fig. 6 compares the 6-bit scheme. In the figure, data point 0 represents five seconds of flight time and is when the attacks are implemented into the system. Each 1000 data points represents an additional five seconds.

In both the 4 and 6-bit representations, the random value attack is detected more often than the unstable zero attack. Data point 2000 represents fifteen seconds of flight and the majority of the detectors that are activated in the unstable zero attack occur on or after this point. The system has already begun to fail at this point. In the random value attack, however, many detectors are activated before failure of the system has begun. The stealth aspect of the zero dynamics attack is shown in the fact that the immune system has a much harder time detecting the attack.

The 6-bit immune system detects the unstable zero attack more often than the 4-bit but not a significant amount. The majority of the detectors are activated after the fifteen second mark. In the random value attack, however, many more detectors are activated before failure occurs. Between thirteen and fourteen and a half seconds there are many more detectors activated. This detection should allow the system enough time to remove the attack and prevent the system from crashing.

Due to the fact that the real system's response did not exponentially grow like that of the simulation, the real system's immune system has a much harder time detecting the attacks. The simulation's immune system almost immediately recognizes the attack since the oscillations grow quickly. In the real system, the changes are much smaller and gradual since the system cannot change angles as quickly as the simulation predicts.

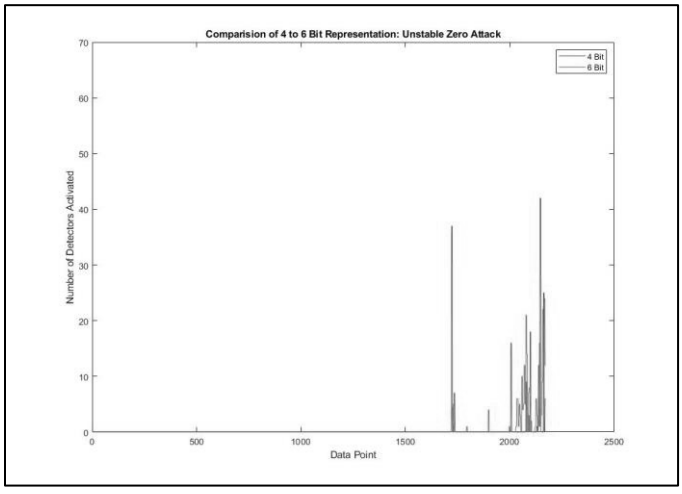

Fig. 5. Detection rate of real system under the unstable zero attack.

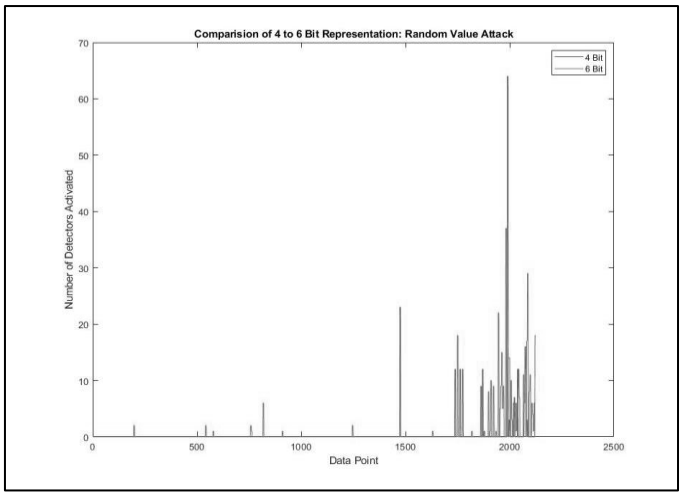

Fig. 6. Detection rate of real system under the random value attack.

VI. Conclusion

The goal of this project is to train an AIS to detect zero-dynamics attacks that are being injected into the commanded roll angle input. The zero-dynamics attacks were developed using a linear model of the Parrot Mambo represented by transfer functions that were found using the system identification toolbox via real test data from the quadcopter. The zero-dynamics attack is implemented into the quadcopter and it is switched on after the quadcopter has stabilized. The authors used the flight test data from the zero-dynamics attack and random value attack tests and ran it through the AIS. The 6-bit scheme is found to perform better at detecting the attacks sooner than the 4-bit scheme. When creating the transfer functions from system identification, the authors found that the linear system does not model the nonlinear quadcopter very well and this could have been the source of many problems faced throughout the project because the zero-dynamics attacks were designed from the transfer functions. The stealth aspect of the zero-dynamics attack that is seen in the linear model is lost when it is transferred to the nonlinear system. A small aspect of stealth is regained, however, due to the fact that the AIS does not detect the unstable zero attack as often or as quickly as the random value attack.

VII. Future work

This project is still ongoing, and one of the future plans is to simulate the entirety of the zero-dynamics attack, from hacking into the drone successfully to injecting the attacks. After extensive reading and testing, it was determined that Bluetooth would not be a viable attack surface. Wi-Fi and 915MHz ISM band are highly favorable over Bluetooth. First, Bluetooth is far more complicated and convoluted than Wi-Fi – the standards themselves are hundreds of pages longer than those for Wi-Fi. Despite its outstanding complexity, though,

Bluetooth is by default very secure. Common Bluetooth exploits such as BlueBorne have all been patched out in recent versions, which our quadcopter runs. This was experimentally verified using an Android phone and a utility produced by the authors of the exploit themselves. Wi-Fi, on the other hand, is very easy to set up insecurely. Almost all Wi-Fi-controlled commercial drones use an open Wi-Fi network, which is relatively easy to exploit. Others use insecure security protocols, or use a default password that can be easily found on the manufacturer's website.

Second, the Bluetooth hacking scene is significantly smaller than that of Wi-Fi. While most Bluetooth exploits throughout history aimed to steal vCards or phone numbers from mobile phones, Wi-Fi exploits are of much greater interest to hacking communities, as they can apply to much more expansive and varied networks, often housing more valuable data. Because of this, there is a higher quantity and greater efficacy of Wi-Fi exploits available online and in bundles, such as Kali Linux. 915MHz and surrounding bands also typically house direct radio instructions between controller and aircraft, which should theoretically be much easier to intercept, and potentially inject as well.

One protocol often used within the 915MHz band is MAVLink. MAVLink communication is unauthorized and unencrypted, meaning that commands can be easily injected or intercepted. An external actor can not only listen to the communications happening between the GCS and the UAV, but also carry out a man-in-the-middle attack by disabling or even hijacking the vehicle [8]. This, as well as exploitation of the Wi-Fi vulnerabilities, can have devastating consequences, as the Confidentiality, Integrity, and Authenticity can all be violated with relative ease by an attacker. Though this method may not be completely stealthy visually, it is a good starting place for attack injection. In an unmanned system, such as that in this study, the AIS would be the first line of defense against a stealthy dynamics attack.

Once a successful hacking method has been acquired, the aim will be for the artificial immune system to be placed online the drone. The malicious user will then use the hacking method to add the attack signal to the commanded value that has been given. The AIS will be used to detect this hacking through the methods described in this paper in real time.

Acknowledgments

This project would not have been possible without the support of the National Science Foundation, Grant no. CNS-1757781. Many thanks go to the foundation for their support of undergraduate research and their role in making this project possible. Other thanks go to Dr. Hever Moncayo for his mentorship, guidance, and encouragement during the process. To the graduate students working in the Advanced Dynamics and Controls Lab, thank you for your support and always checking in on the team.

References

[1] J. Kaneshige and K. Krishnakumar, "Artificial immune system approach for air combat maneuvering," Proc. SPIE 6560, Intelligent Computing: Theory and Applications V, 656009, 2007.

[2] M. Perhinschi, H. Moncayo, B. Wilburn , J. Wilburn, O. Karas and A. Bartlett , "Neurally-augmented immunity-based detection and identification of aircraft sub-system failures," The Aeronautical Junary (1968), 2014.

[3] H. Moncayo, "AIS Discrete Data Representation".

[4] H. Jafarnejadsani, H. Lee, N. Hovakinmyan and P. Voulgaris, "Dual-rate L1 adaptive controller for cyber-physical sampled-data systems," IEEE 56th Annual Conference on Decision and Control (CDC), 2017.

[5] H. Jafarnejadsani, "Robust adaptive sampled-data control design for MIMO systems: Applications in cyber-physical security," 2018.

[6] J. A. Gross and T. Staff, "Iranian UAV that entered Israeli airspace seems to be American stealth knock-off," Times of Israel, 2018.

[7] J. Verberne, *Development of Robust Control Laws for Disturbance Rejection in Rotorcraft UAVs,* (Master's Thesis) Embry-Riddle Aeronautical University Scholarly Commons, 2019.

[8] M. A. Joseph, "Vulnerability Analysis of the MAVLink Protocol for Command and Control of Unmanned Aircraft," 2014.