SECURING THE ADS-B PROTOCOL FOR ATTACK-RESILIENT MULTIPLE UAV COLLISION AVOIDANCE

by

Zachary P. Languell, B.S.

A thesis submitted to the Graduate Council of
Texas State University in partial fulfillment
of the requirements for the degree of
Master of Science
with a Major in Computer Science
May 2019

Committee Members:

Qijun Gu, Chair

Xiao Chen

Mina Guirguis

FAIR USE AND AUTHOR'S PERMISSION STATEMENT

Fair Use

Duplication Permission

ACKNOWLEDGEMENTS

I would like to thank Dr. Qijun Gu, my thesis and research advisor, for his supervision, contributions, and mentorship. His guidance and support were crucial to the success of this research. I would also like to thank Dr. Xiao Chen and Dr. Mina Guirguis for supporting this work and participating on my committee. Finally, I would like to thank my family and friends whose patience and understanding through this endeavor has allowed me to pursue my education.

TABLE OF CONTENTS

Page

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

ABSTRACT

The Automatic Dependent Surveillance-Broadcast (ADS-B) protocol is being adopted for use in unmanned aerial vehicles (UAVs) as the primary source of information for emerging multi-UAV collision avoidance algorithms. The lack of security features in ADS-B leaves any processes dependent upon the information vulnerable to a variety of threats from compromised and dishonest UAVs. This could result in substantial losses or damage to properties. This research proposes a new distance-bounding scheme for verifying the distance and flight trajectory in the ADS-B packets from surrounding UAVs. The proposed scheme enables UAVs or ground stations to identify dishonest UAVs and avoid collisions. The scheme was implemented and tested in the SITL (Software In The Loop) simulator to verify its ability to detect dishonest UAVs. The experiments showed that the scheme achieved the desired accuracy in both flight trajectory measurement and attack detection.

I. INTRODUCTION

The Automatic Dependent Surveillance-Broadcast (ADS-B) is a surveillance technology used by aircraft to share position and navigation information with surrounding aerial vehicles and ground elements. Starting January 1, 2020, aircraft must be equipped with ADS-B Out to fly in most controlled airspace [10]. The Federal Regulations mandating this technology will have a direct impact on how new and developing systems may implement functionality, giving it focus on collision avoidance algorithms utilizing this service. It will also influence existing work on ADS-B based collision avoidance algorithms[21]. This is of great concern and interest to the unmanned aerial vehicle (UAV) community as well [31, 28, 21].

Commercial use of ADS-B for UAVs is rising as more devices become available [33, 2]. These systems have the potential to improve air safety. Unfortunately, using ADS-B data alone to make automated in-flight decisions is unreliable due to the wide range of threats associated with the ADS-B Protocol[32, 7, 24]. The broadcasts are plain-text and lack any method of authenticating the broadcaster. This means the protocol is susceptilbe to a wide range of common wireless communication vulnerabilities including injection, spoofing, modification, and jamming among others.

There have been studies on various security implementations for authenticating ADS-B vehicles[35, 37]. Authentication does not prevent a compromised system from being exploited to manipulate other aircraft using fraudulent packet data. This is why it is crucial that data, such as location and velocity, be verified in the ADS-B security solution. Distance-bounding is one such concept that was initially designed for verifying distance between two agents.

Typical distance-bounding protocols measure the time between challenges and responses to determine if an agent exceeds a distance threshold. Several rounds of single-bit challenges are sent, and the time it takes to receive a single-bit response is directly related to the physical distance between them. This is reliable when the distance is small and the propagation speed is deterministic. If UAV interaction is to

adopt this concept, it must consider large distances, fast-moving UAVs, GPS error, processing time variations, and potentially lost packets.

This work introduces a new distance-bounding protocol to be used as a supplementary component to ADS-B. The new protocol uses a multi-point distance-bounding session for mobile agents allows the system to detect and filter fraudulent ADS-B messages in real time while adding minimum communication and computational overhead. A verifying agent can estimate future locations from the extracted ADS-B navigation information and perform a lightweight distance-bounding session with the prover at each of these predicted locations. The protocol can be used to detect distance fraud or suspicious prover activity.

This research demonstrates that utilizing ADS-B and multi-point distance-bounding, a system can reliably detect and filter fraudulent messages in the vicinity and allow UAVs to make maneuvering decisions regarding collision avoidance with a confidence.

Contributions from this thesis include: (1) Analysis of multi-point distance-bounding protocol; (2) implementation of multi-point distance-bounding in SITL simulation

The rest of the thesis is organized as:

Chapter II discusses the related security works with this thesis.

Chapter III provides background for ADS-B and distance-bounding protocols.

Chapter IV describes the new distance-bounding and how it uses ADS-B

Chapter V presents an analysis on the impact of various errors and calculations.

Chapter VI shows the implementation of the protocol in the simulation environment.

Chapter VII presents the experiment settings and the evaluation of the results.

Chapter VIII discusses the conclusion and future works of the research.

II. RELATED WORKS

Strohmeier et al. survey[32] was performed on the security and future of the ADS-B protocol. It identifies, categorizes, and evaluates a variety of vulnerabilities and possible security implementations.

Vulnerabilities

Message Deletion

Message deletion is one category of attacks against ADS-B messages. The desired outcome of these attacks is to cause a receiving party to discard or not receive the message. The most feasible form of this attack is an attack that disrupts enough of the message to fail the parity check, which is at most 5 bits[23, 32]. This vulnerability is often used in conjunction with another vulnerability to carry out an attack.

Message Modification

Message modification is another category of attacks against the ADS-B protocol. The attacker tries to modify the message between the sending and receiving parties. This is accomplished through overshadowing or bit-flipping[36, 23]. Overshadowing requires a stronger signal than the sending party relative to the receiver. Bit-flipping requires precise synchronization to superimpose the attackers signal over the victim signal. Overshadowing is more feasible when dealing with moving aircraft.

Message Injection

Message injection is a serious threat to ADS-B systems due to a lack of authentication. An attacker can create ghost aircraft, aircraft that do not exist, by flooding ADS-B messages into an airspace[7]. The injections can be singular or multiple depending on the goal of the attacker. Multiple ghost aircraft can be used to overwhelm a system, while single ghost may be used to cause honest surrounding aircraft to change

trajectory or flight plan. It was proven [23] the limit of ghost aircraft flooding is the bandwidth of the ADS-B channel.

Signal Jamming and Eavesdropping

Although a concern for ADS-B, these are threats common to all wireless communications and against the form of communication rather the protocol. Since ADS-B is in plain-text, eavesdropping is more of a concern to facilitate replay attacks which combine eavesdropping with injection.

Security Features

Secure Broadcast Authentication

One way to secure the protocol is through verifying the identity of the broadcasting agent. Authentication allows any receiving devices to verify the sending device, but not necessarily the information being transmitted. This category of defenses attempts this in a few different ways.

1. Public Key Cryptography

   PKI is a very common authentication process well researched and described in security textbooks. There are numerous difficulties implementing this in ADS-B including: key distribution, communication overhead, and lack of centralized and available certificate authorities.

2. Random Frequency Hopping

   Randomly hopping communication frequencies is one way to prevent eavesdropping or jamming attacks. Both parties would need to know the pattern to stay in synchronization, so this would be difficult to keep secret.

Secure Location Verification

This category of defense mechanisms is designed to verify the claimed physical location of the prover instead of the identity of them. Authentication by itself is

insufficient to secure the protocol, a compromised aircraft could distribute authenticated but sabotaged ADS-B messages. This makes it critical to verify the location of the transmitting device. There are several different ways this can be accomplished, but some require additional participants or hardware which may be too cumbersome to implement on a wide-scaled system already in place.

1. Multilateration

   This method requires several antennas to receive the same signal. Measuring the difference in time-of-arrival between them, a position can be estimated by finding the intersection. This operates like GPS, and is currently used as a popular solution for ground stations making it potentially adaptable to ADS-B[29]. Received signal strength could also be used, but is vulnerable to an attacker adjusting the strength of their signal in directional broadcasts.

   Wide area multilateration is another commonly suggested solution to verifying ADS-B messages. This concept increases the range problem associated with multilateration, but still would require communication between all participants.

2. Group Verification

   Group verification, or group concept, involves the use of multiple parties to triangulate the location of a suspicious or untrusted aircraft[27]. The additional participant must be a trusted party and not assisting the adversary. It very closely resembles multilateration. The group concepts largest flaw is the requirement of several other aircraft. In dense airspaces this is more feasible, but UAVs may be operating in sparse airspaces. It suffers from the same problems as multilateration as well, including extensive communication overhead with other aircraft.

3. Kalman Filtering

   Already used in GPS systems, Kalman Filtering, or Bayesian filtering, [12] can predict future states in noisy data sets. The process happens in two stages, a

prediction then update stage. It is useful as a tool to another method such as plausibility checks and multilateration. A frog boiling attack[6] can defeat Kalman filtering by blocking correct signal and supplying slightly skewed. If this process is done slowly over time, the filter will not be able to distinguish an attack.

4. Distance-bounding

   Distance-bounding is a technique of determining the physical distance between two agents by measuring time between responses. It is used in a variety of real world applications such as card or badge readers. The core concept of distance-bounding is a reliable communication medium to measure response times across. Distance-bounding was created to defeat distance fraud attacks, shown in Chapter III.2.2. Different versions of the protocol have developed to defeat certain types of attacks or meet different situational needs. An overview of other distance-bounding protocols is discussed in Chapter III.2.3.
   Unlike some of the above concepts, such as multilateration and group concept, distance-bounding does not require communication with any agents other than the one attempting to be verified. This is particularly desirable with UAVs since they may be operating in sparse airspace. Hardware requirements is another reason distance-bounding is particularly fitting for supplementing ADS-B in UAVs. A small processing device could easily be attached to the UAV. It does not require a large or elaborate modification to the device.

   This work greatly expands upon the distance-bounding concept in Chapter III.2.

5. Data Fusion

   Data fusion is using any combination of information to verify each other. Several works[22, 15] have proposed methods of doing this procedure, but the sensors involved typically are not available for UAVs. Any solution could take advantage of this concept and further the confidence of the solution so long as the fusion does not introduce a new attack vector on the solution.

III. BACKGROUND

ADS-B

Overview

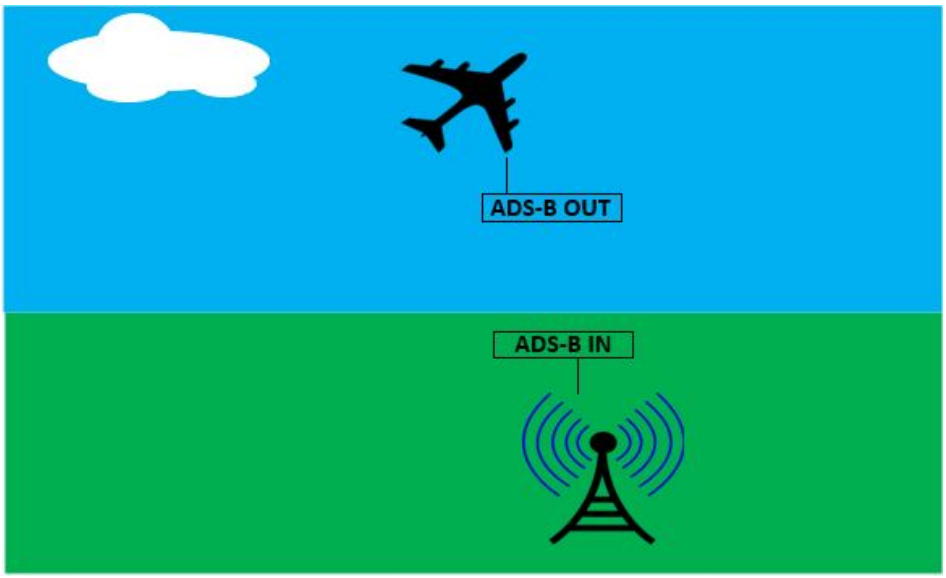

Figure III.1: ADS-B In and Out with Ground Station

Advanced Dependent Surveillance - Broadcast is the NextGen surveillance solution designed to replace secondary radar and improve safety in the air. The system is compromised of two capabilities, ADS-B Out and ADS-B In. ADS-B Out is mandated by January 1, 2020 in designated airspace throughout the United States and already mandated in some European airspace. An example of standard ADS-B Out and In is shown in III.1 where an aircraft communicates with a ground station. ADS-B Out automatically broadcasts information regarding the aircraft and its intent in omni-directional 1090Mhz/978Mhz UAT transmissions. The system is dependent upon the navigation systems for its information. These broadcasts are broadcast multiple times a second throughout the flight and can be used by both other aircraft and ground-stations alike.

The counter part to ADS-B Out subsystem is the ADS-B In subsystem, the ground station in III.1. This facilitates reception and demodulation of nearby ADS-B Out

broadcasts. This subsystem has no current mandates announced, but it is an integral part of the collision avoidance functionality desired in aircraft and UAVs. Adaptations of these devices designed for small UAVs are already on the market, making ADS-B a viable solution. The information gathered from nearby broadcasts can be used to dynamically adjust flight plans even when operating in way-point mode without ground-station communication. This is a highly desired quality of the collision avoidance problem as the airspace becomes more crowded.

Key Packet Specifications

All ADS-B messages use the Downlink Format- 17/18 (DF17/18), Figure III.2. DF17 is a 1090 Extended Squitter while DF18 is a 1090 Extended Squitter, supplementary. DF18 implies Non-Transponder-Based ADS-B Transmitting Subsystems and Traffic Information Service-Broadcast (TIS-B) Transmitting equipment, alerting others that the message comes from equipment that cannot be interrogated.

ADS-B / TIS-B Message Format					
Bit	1 --- 5	6 --- 8	9 --- 32	33 --- 88	89 --- 112
Field	DF Downlink Format	CA Capabilities	AA ICAO Address	ME ADS-B Message Field	PI Parity/Interrogator

Figure III.2: ADS-B Message Structure: Downlink Format 17/18

The first five bits of the broadcast identifies the Downlink format, followed by a three-bit capability field. An ICAO address field of 24-bits identifies the transponder sending the signal, and then 56-bits are used for the data. An unique ICAO address is assigned to each Mode-S transponder of an aircraft. The ADS-B uses a cyclic redundancy check to validate the correctness of the received message, where the last 24-bits are the parity bits.

There is a variety of message types at the systems disposal, and are identified by the type code representing the first five bits of the data field. These types range from routine messages, like identification and position, to specialized and circumstantial ones, such as target state and status. A list is shown in III.1. None of the information in

these messages is encrypted. The system will prioritize, from highest to lowest, non-event driven messages as follows: position message, airborne velocity message, aircraft identification message.

Table III.1: ADS-B Type Codes

Type Code	Message Content
1-4	Aircraft identification
5-8	Surface position
9-18	Airborne position (Barometric)
19	Airborne velocity
20-22	Airborne GNSS
23-27, 30	Reserved
28	Aircraft status
29	Target state and status information
31	Aircraft operation status

In this work, we focus on a few important ones that pertain to all collision avoidance scenarios.

Aircraft Identification ME Field					
Bit	33 --- 37	38 --- 40	41 --- 46	47 --- 82	83 --- 88
Field	Type Code	ADS-B Emitter Category	Identification Characater #1	...	Identification Character #8

Figure III.3: ADS-B Aircraft Identification Message Structure

The aircraft identification message includes the ADS-B emitter category, which gives information about the size and type of aircraft. It also includes 48 bits for the callsign or identity. This message is sent on average once every five seconds while airborne. This information is never cleared out despite no updates from navigation, and this message never terminates its broadcast. The layout of the information can be seen in Figure III.3.

The airborne position message contains altitude and a modified latitude/longitude encoded value. This lat/long encoding requires an odd and even frame sequence that

Airborne Position ME Field								
Bit	33 --- 37	38 --- 39	40	47 --- 82	53	54	55 --- 71	72 --- 88
Field	Type Code	Surveillance Status	NIC Supplement - B	Altitude	Time	CPR Format	CPR Encoded Latitude	CPR Encoded Longitude

Figure III.4: ADS-B Aircraft Position Message Structure

are indicated by a flag bit in the message. This message is sent on average twice every second while airborne. All bits except for altitude and surveillance status are cleared after 2 seconds with no update from the navigation system. After 60 seconds without position or altitude information from the navigation system, the transponder will terminate airborne position message broadcasting.

Airborne Velocity Types 1 & 2 ME Field															
Bit	33 --- 37	38 --- 40	41	42	43 --- 45	46	47 --- 56	57	58 --- 67	68	69	70 --- 78	79 --- 80	81	82 --- 88
Field	Type Code	Subtype	Intent Change Flag	Reserved	NACv	E/W Direction	E/W Velocity	N/S Direction	N/S Velocity	Vert Rate Source	Vert Rate Sign	Vert Rate	Reserved	Diff from Baro Alt Sign	Diff from Baro Alt

Airborne Velocity Types 3 & 4 ME Field															
Bit	33 --- 37	38 --- 40	41	42	43 --- 45	46	47 --- 56	57	58 --- 67	68	69	70 --- 78	79 --- 80	81	82 --- 88
Field	Type Code	Subtype	Intent Change Flag	Reserved	NACv	Heading Status	Heading	Airspeed Type	Airspeed	Vert Rate Source	Vert Rate Sign	Vert Rate	Reserved	Diff from Baro Alt Sign	Diff from Baro Alt

Figure III.5: ADS-B Aircraft Velocity Message Structures

The velocity message has different subtypes and provides either cardinal direction velocity or heading and airspeed. Both include vertical rate as well as a navigation accuracy estimate. This message is sent on average twice every second while airborne. All 56 bits are cleared after 2.6 seconds with no update from the navigation system and the message broadcast is terminated.

Distance-bounding

Overview

Distance-bounding is the process of timing an interaction between two systems across a medium, a verifying party known as Verifier and a proving party known as Prover. Determining the distance between them is based on the speed of the signal propagation across the medium plus the processing time of the Prover. The RF

implementation[14][1] of distance-bounding relies on the fact RF waves travels near the speed of light. The Round-Trip-Time distance formula $d = \frac{c \cdot (t_m - t_d)}{2}$ where c is the propagation speed of the medium, the total RRT is t_m, and t_d is the time the prover takes to calculate the response. The distance estimate calculated here is most effective when t_d and d are consistent, because the estimate error is predictable.
This process is repeated for for several times during what is known as a fast phase. There are several variations on this process, some works include what is called a void challenge[26]. A void challenge means no challenge is sent for that interaction. Another method seen in [9] and [20] involves each challenge can mean respond or wait a predetermined amount of time. In [20] the prover still responds after the wait while [9] does not. These variations and others help defeat particular types of attacks.
Once the interactions are completed, the verifier will check that the responses are correct and do not exceed T_{max}, a determined value for the maximum distance the verifier wants to allow. Distance-bounding is common in a wide variety of systems, such as credit card readers and security badge readers. The process is not completely safe, there are a variety of vulnerabilities designed to defeat or improve the chances of defeating a distance-bounding process.

Threat Models

There are attacks designed to defeat distance-bounding protocols: Distance Fraud, Mafia Fraud, Terrorist Fraud, and Distance Hijacking.

1. Distance Fraud

   A dishonest prover or an adversary claims to be somewhere in the verifiers vicinity. This is a broad category that can define attacks not fitting of a more specific one. The proceeding types could be considered variations or sub-types of distance fraud.

2. Mafia Fraud

   In a Mafia fraud attack[25], an adversary exists between a honest prover and a

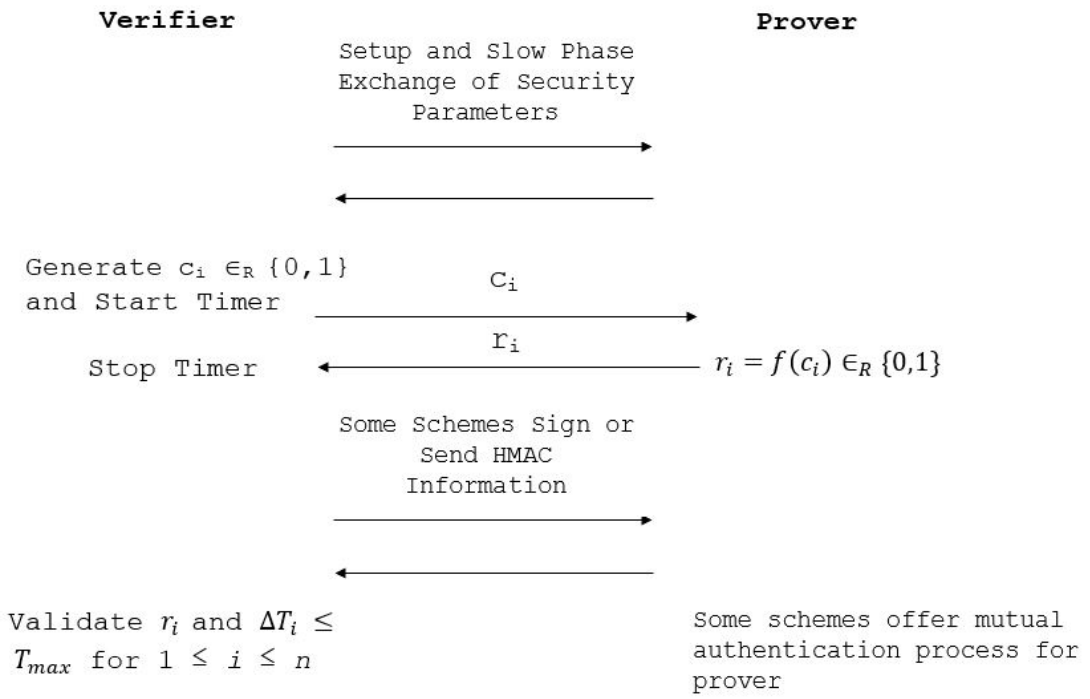

Figure III.6: Generalized Distance-bounding

verifier. The attack closely resembles Man-in-the-Middle or Relay attacks. The adversary tries to make the distance between these two seem shorter than it is in reality. An example of this threat, an adversary uses an RF reader to pick up your badge signal and send the communications to another location where a transmitter is being placed near the verifier. In distance-bounding, the time it takes for the adversary to send these transmissions back and forth would exceed T_{max}.The probability of success varies on the medium and distance-bounding protocol[18].

3. Terrorist Fraud

   An adversary uses a dishonest prover to conduct the attack, but it must be in such a way that it does not give any assistance in future attacks. Terrorist Fraud attacks[13] are considered thwarted when assisting an adversary would reveal the dishonest provers long term secret or key. If the dishonest prover can assist an adversary without revealing any damaging or long term secrets, then the protocol is considered vulnerable.

4. Distance Hijacking

   Distance Hijacking is a recently discovered attack procedure where a dishonest prover uses an honest prover by hijacking their verification phases[8]. In this attack, an attacker will take advantage of an honest prover without their assistance or consent. This often means hijacking the interaction between an honest prover and verifier during the fast phase.

Overview of Existing DB Schemes

Distance-bounding protocols consist of three main phases: Initialization , protocol, and a final decision. The initialization includes agreeing upon security or protocol parameters and in some protocols other functions for authentication. A survey[3] provides a comparison of these protocols in both functionality and security.

1. BRANDS AND CHAUM'S PROTOCOL

   This protocol[5], designed in 1993, is the first distance-bounding protocol. It was created to defeat mafia and distance fraud threats.

2. CAPKUN, BUTTYAN, AND HUBAUX'S PROTOCOL

   The mechanics of this protocol[34] allow for mutual authentication between parties. Both participating parties act as a verifier and prover during the protocol. Each exchange is a challenge from the other party that can be verified. It operates in similar fashion to the BC protocol in all other regards.

3. HANCKE AND KUHN'S PROTOCOL

   This protocol[14] introduced in 2005 was designed for use with RFID technology, making it a good candidate for verifying ADS-B information. There is no final signature feature. The lightweight nature of this protocol makes it fast and efficient for use with ADS-B. It is susceptible to terrorist fraud, as the dishonest prover can provide the response sequences without revealing any long term secret.

4. MUNILLA AND PEINADO'S PROTOCOL

   The unique feature of this protocol involves the use of void challenges. Each challenge could be 0, 1, or void. In the case of void, no challenge is sent to the proving party. It is effective against pre-ask attack models and can be adapted well for RFID as shown [26].

5. KIM, AVOINE, KOEUNE, STANDAERT, AND PEREIRA'S PROTOCOL

   Also known as the Swiss-knife distance-bounding protocol, this protocol optimizes many of the qualities of previous protocols including mutual authentication. The author states, "resists against both mafia fraud and terrorist attacks, reaches the best known false acceptance rate, preserves privacy, resists to channel errors, uses symmetric-key cryptography only, requires no more than 2 cryptographic operations to be performed by the tag, can take advantage of precomputation on the tag, and offers an optional mutual authentication"[17].

6. AVOINE AND TCHAMKERTEN'S PROTOCOL

   The Avoine and Tchamkerten protocol[4] is a generalized form of the Hancke and Kuhn RFID protocol. It uses binary trees, and in its most basic state reduces to the HK protocol.

7. YUM, KIM, HONG, AND LEE'S PROTOCOL

   Another mutual authentication protocol, YKHL checks for collisions during each round. In the case of a collision, the party enters protection mode and proceeds to send random bits for the following rounds. It has an adjustable acceptance rate version[16].

8. PUF-BASED PROTOCOLS

   This protocol uses Physically Unclonable Functions (PUFs) to replace secrets, helping defeat terrorist fraud[19] attacks. These types of functions are based on unique physical properties of the device that are a result of how it was manufactured.

IV. PROPOSED DB IN ADSB

Threat Model

The addition of distance-bounding to ADS-B interactions is not without its own set of threats. In this research we consider some threat models but it is not all inclusive. The core of the threat is a compromised or dishonest prover that commits distance fraud against an honest verifier. The attacker will always claim to be closer to the verifiers location than they actually physically located. The goal of the attacker is to force the victim into taking collision avoidance maneuvers, causing the victim to alter its flight plan.

An intelligent threat could use the victims automatic collision avoidance response to approaching messages to control or manipulate a victim into altering its trajectory unnecessarily or into unsafe conditions. This can be disruptive to both the victim and other surrounding aircraft.

Multi-Point DB Model

RF based distance-bounding is used in many current systems today such as debit and credit card verification schemes, badge protected building access, and others. These systems involve a static verifying device and a near static prover. They interact when held very close to each other, sometimes even in contact with another. Original analysis of this model was not suited for the UAV ADS-B verification process. Detection needs to occur at a significantly larger distance than traditional distance-bounding operations, leaving the UAVs to take appropriate actions in the scenario collision is possible. Fast-moving UAVs is another concern when porting distance-bounding to this type of problem. Changes in distance during the protocol are not part of the intended model. When considering a supplementary component to ADS-B, the broadcast message has some significant challenges to overcome as well:

1. There is no time-stamp in the ADS-B message when it was generated.

2. GPS navigation equipment has an error range, so the location information isn't exact.

3. Some data from velocity or position messages may be up to two seconds old.

With all these challenges to overcome, the standard method of distance-bounding is incapable of providing an accurate way to filter bad ADS-B messages. The protocol would be susceptible to high numbers of false positive fraud detection, because the propagation medium is noisy resulting in lost challenges and a fluctuating processing time combined with imprecise navigation information. The allowable limit of failed challenges would either be too high to filter out bad messages or too low to allow realistic message variation. Some work has been done on algorithms for adjustable acceptance rates [38], but adjusting the rate is not a solution to the challenges associated with UAVs.

If an attacker or compromised UAV was to create ADS-B messages containing fraudulent information, they could use the allowable variance in the processing time and fault tolerance of the acceptance rate to deceive a verifier. In order to deal with this, the protocol needs to be able to mitigate both failed challenges and processing time noise manipulation by an intelligent attacker.

The new multi-point distance bounding protocol is a concept designed to meet the challenges introduced by two mobile agents using ADS-B. A verifier can predict multiple points on the proving UAV agents flight path after receiving an ADS-B Out broadcast. Like traditional distance-bounding, the core of the protocol is a fast exchange of bits between a verifiying and proving agent. However, this will be down in several rounds with a random amount of delay between each round corresponding to the time it will take the prover to get to a predicted location the verifier has selected. The prover does not know about these locations or delays, only how many will occur. Using this process, the verfier can check the correctness of the ADS-B information from the prover.

Procedure:

Starting with a prover's ADSB packet, the verifier obtains the position $\boldsymbol{p}_0$ and velocity

$\boldsymbol{v}$ of the prover.

1. The verifier chooses a set of random points $t_1, t_2, ..., t_n$ to perform distance bounding.

2. At each t_i, the verifier measures $\check{\tau}_{i,j}$ m times, i.e. $j = 1, 2...m$. For example $m = 16$ times, which means two bytes

3. The verifier estimates the average $\hat{\tau}_i$, $\hat{t_p}$, $\hat{d}_i$.

Static Verifier

Verifier is stationary at the origin of a coordinate system. It could be a ground station or a hovering UAV. The prover is a mobile UAV as shown in IV.1. This model is emulated in the simulation where the verifier is a stationary UAV.

Prover's velocity is $\boldsymbol{v} = [v_x, v_y, v_z]'$, a constant vector. $v^2 = \boldsymbol{v}'\boldsymbol{v} = v_x^2 + v_y^2 + v_z^2$.

Prover's position at t_i is $\boldsymbol{p_i} = [x_i, y_i, z_i]' = \boldsymbol{p_0} + \boldsymbol{v}(t_i - t_0)$

Prover's distant to verifier at t_i is $d_i = |\boldsymbol{p_i}| = \sqrt{x_i^2 + y_i^2 + z_i^2}$.

$d_i^2 = \boldsymbol{p_i'}\boldsymbol{p_i} = (\boldsymbol{p_0} + \boldsymbol{v}(t_i - t_0))'(\boldsymbol{p_0} + \boldsymbol{v}(t_i - t_0)) = d_0^2 + 2\boldsymbol{v}'\boldsymbol{p_0}(t_i - t_0) + v^2(t_i - t_0)^2$.

Round-trip time delay between prover and verifier is

$\tau_i = \frac{2}{c}d_i + t_p = \frac{2}{c}\sqrt{x_i^2 + y_i^2 + z_i^2} + t_p$

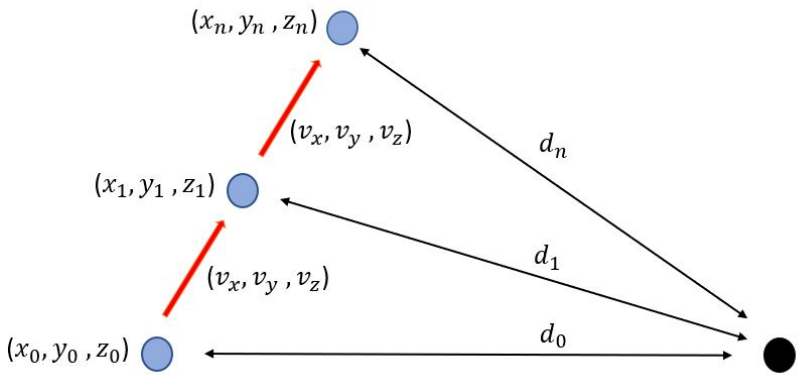

Figure IV.1: Distance-bounding with Mobile Prover and Static Verifier

Mobile Verifier

In many cases, the verifier will not be stationary but mobile as well. This model provides prover and verifier are mobile UAVs, and demonstrates the model can be reduced to an equivalent static verifier each round.

Verifier's velocity is $\boldsymbol{v_v} = [v_{vx}, v_{vy}, v_{vz}]'$, a constant vector.

$v_v^2 = \boldsymbol{v_v'v_v} = v_{vx}^2 + v_{vy}^2 + v_{vz}^2$.

Prover's velocity is $\boldsymbol{v_p} = [v_{px}, v_{py}, v_{pz}]'$, a constant vector.

$v_p^2 = \boldsymbol{v_p'v_p} = v_{px}^2 + v_{py}^2 + v_{pz}^2$.

Verifier's position at t_i is $\boldsymbol{p_{vi}} = [x_{vi}, y_{vi}, z_{vi}]' = \boldsymbol{p_{v0}} + \boldsymbol{v_v}(t_i - t_0)$

Prover's position at t_i is $\boldsymbol{p_{pi}} = [x_{pi}, y_{pi}, z_{pi}]' = \boldsymbol{p_{p0}} + \boldsymbol{v_p}(t_i - t_0)$

Prover's distant to verifier at t_i is

$d_i = |\boldsymbol{p_{pi}} - \boldsymbol{p_{vi}}| = |(\boldsymbol{p_{p0}} - \boldsymbol{p_{v0}}) + (\boldsymbol{v_v} - \boldsymbol{v_p})(t_i - t_0)|$.

Let $\boldsymbol{p_0} = \boldsymbol{p_{p0}} - \boldsymbol{p_{v0}}$ and $\boldsymbol{v} = \boldsymbol{v_v} - \boldsymbol{v_p}$. Then, $d_i = |\boldsymbol{p_0} + \boldsymbol{v}(t_i - t_0)|$.

$d_i^2 = (\boldsymbol{p_0} + \boldsymbol{v}(t_i - t_0))'(\boldsymbol{p_0} + \boldsymbol{v}(t_i - t_0)) = d_0^2 + 2\boldsymbol{v'p_0}(t_i - t_0) + v^2(t_i - t_0)^2$.

Round-trip time delay between prover and verifier is $\tau_i = \frac{2}{c}d_i + t_p$.

Complete Protocol

The Multi-point distance-bounding protocol includes five steps; Authentication, Setup, Bit Exchange, Verification, and Attack Detection. These steps are outlined in IV.2, and each step is explained in detail below.

Authentication

An authentication process is necessary to transmit a secret symmetric key used in the setup phase. This needs to occur before the distance-bounding can begin. This work adopts the authentication methods in existing secure distance-bounding protocols, for example a Diffie-Hellman exchange. The security of any secrets or a key must not compromised during this exchange. Both parties should know the size of the secret key before the exchange.

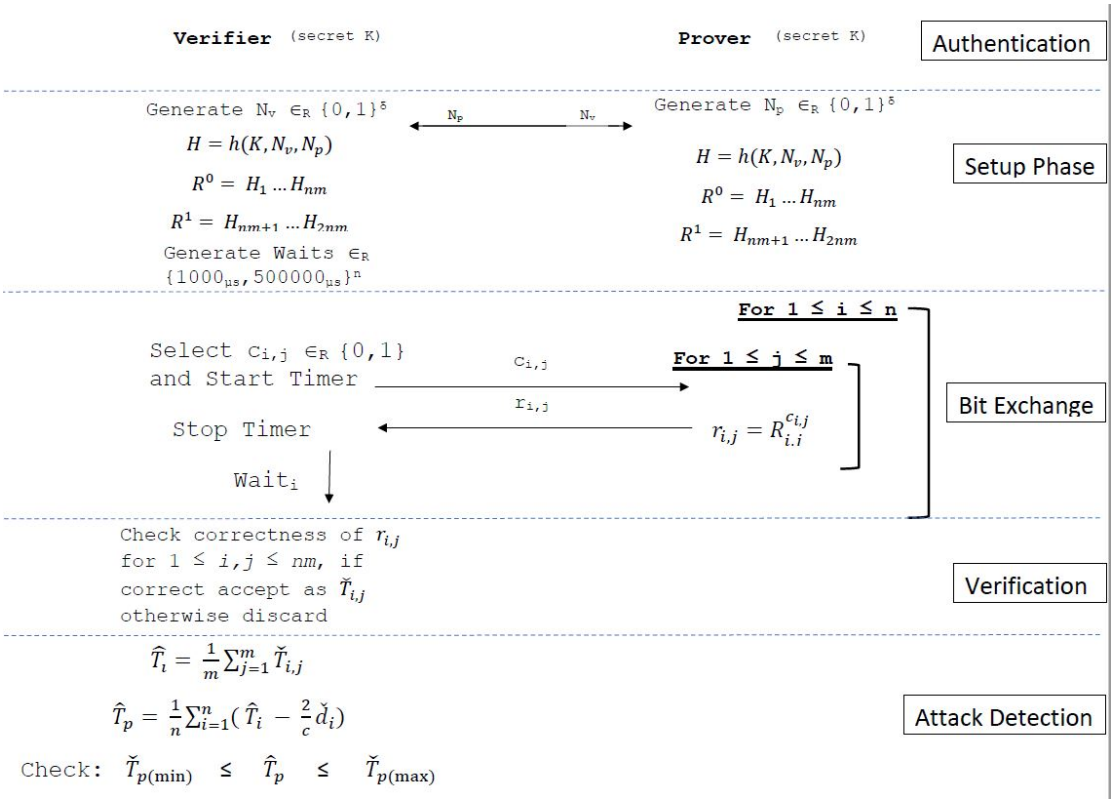

Figure IV.2: Protocol Overview with Boundary Checking

Identity Based Encryption (IBE) using elliptic curve is another potential option for this authentication process. ICAO addresses are unique identifiers associated with Mode-S transponders used in ADS-B. These are a potential candidate for identity.

Setup

Before setting up the protocol, several parameters should be agreed upon by the verifier and prover:

1. Number of rounds n and challenges m
2. Size of nonces, N_v and N_p
3. Hash function with an appropriate output size

The setup phase, also known as the slow phase, is used to exchange nonces N_v and N_p. During this time, each party also sets up two bit sequences of equal length to the total number of challenges, R^0 and R^1. The verifier will randomly generate n wait times to be used between challenges during the fast phase. The wait times are between 1,000

and 500,000 microseconds. Once this information is computed, the fast phase can begin.

Random Bits Exchange

After waiting a randomly generated amount of time, the verifier begins the first round by generating a random challenge bit, c_i, j and sending it to the prover. The prover will respond immediately with one-bit response, r_i, j, from the corresponding bit sequence, R^0 or R^1, determined by c_i. This transaction between them occurs m times before the round, the inner for loop of IV.2, is completed. After the round is complete, the verifier waits according to the generated time, $Wait_i$, for that round before beginning the next round. Once all rounds are completed, the outer for loop of IV.2, the verifier will validate the responses in verification phase.

Verification

The verifier will compare the correct bit sequence with the received prover bit sequence. There may be a small number of failed challenges due to interference on the medium, so the protocol needs to have a small fault tolerance. Increasing the allowable failed challenges will reduce the protocols ability to filter noise, so it should remain as low as possible.

The original distance-bounding scheme uses this metric plus a check against the maximum allowable time to determine if the prover is legitimate. This is a distinct difference in the protocol from this work and other distance-bounding protocols. Challenges are kept despite exceeding an initial time estimate so long as they were correct. Any failed challenges should be removed from the sequence, making sure to note the number of failed challenges for each round. If the total number of failed challenges exceeds a pre-determined threshold, then the session fails. Otherwise, the successful challenges move forward into the attack detection phase.

Attack Detection

The average, $\hat{\tau}_i$, is obtained for each round using the successful challenges. Failed challenges were removed in the previous phase and should not be added into the average.

$\hat{\tau}_i = \frac{1}{m} \sum_{j=1}^{m} \check{\tau}_{i,j}$

From the ADS-B message, an approximated response time for the distance at each round can be calculated. The provers claimed velocity is extracted from the message, and a new position is predicted from the difference between time receiving the message and conducting the round. Every challenge for the round can be compared to this value since the change in distance between challenges is trivial compared to the change in rounds. The estimated distance from the ADS-B message, $\check{d}_i$, subtracted from the average obtained for each round is the corresponding rounds processing time. Averaging them provides the mean processing time for the session.

$\hat{T_p} = \frac{1}{n} \sum_{i=1}^{n} (\hat{\tau}_i - \frac{2}{c} \check{d}_i)$

The protocol makes the initial assumption that the ADS-B message is valid and all information in it is within standard GPS error. This means any distance calculation will appear accurate when calculated using the processing time obtained above in the following formula:

$\hat{d}_i = \frac{c}{2} (\hat{\tau}_i - \hat{T_p})$

However, it is the processing time used to determine whether the ADS-B information is accurate. A processing time lower bound and deviation can be calculated for any device if the computations executed on it are deterministic. In the scenario an attacker is trying to claim a physical location closer to the verifier, $\hat{T_p}$ will be the container for that error.

The verifier can estimate the mean of the processing time plus several deviations to create a boundary that an attacker cannot defeat with a distance fraud attack. Estimation calculations are shown in the next in Chapter V.

This is not the only possible method of detection. Besides a boundary check, the

verifier can analyze several sessions to determine if an attack is occurring. An attacker simulating fake movement will have difficulty maintaining a consistent processing time. Unstable increasing/decreasing processing time is another indication that an attack may be taking place.

V. ANALYSIS

Variable notations

Actual value: $\boldsymbol{p}_i$, $\boldsymbol{v}_i$, d_i, t_p, τ_i, where round $i = 1, 2...n$

Estimated value: $\check{\boldsymbol{p}}_i$, $\check{\boldsymbol{v}}_i$, $\check{d}_i$, $\check{t}_{p_{i,j}}$, $\check{\tau}_{i,j}$, where round $i = 1, 2...n$, bit $j = 1, 2...m$.

Calculated value: $\hat{\boldsymbol{p}}_i$, $\hat{\boldsymbol{v}}_i$, $\hat{t_{p_i}}$, $\hat{d}_i$, $\hat{\tau}_i$

Noise and Error

First consider the noise in the individual variables. The ADS-B information received from the message will have GPS error from the navigation. This will be a static offset for the initial position, but the velocity error will increase as the session duration increases.

$\tau_i = \frac{2}{c} d_i + t_p + \epsilon_i = \frac{2}{c}\sqrt{v^2(t_i - t_0)^2 + 2\boldsymbol{v}'\boldsymbol{p_0}(t_i - t_0) + d_0^2} + t_p$

Noise in ADS-B, not changing over i and j:

$\check{\boldsymbol{p}}_\mathbf{0} = \boldsymbol{p_0} + \boldsymbol{\epsilon_p}$ $\check{\boldsymbol{v}} = \boldsymbol{v} + \boldsymbol{\epsilon_v}$

Noise of t_p at i and j:

$\check{t}_{p_{i,j}} = t_p + \epsilon_{ti,j}$

Noise in measured values:

$\check{\boldsymbol{p}}_i = \boldsymbol{p}_0 + \boldsymbol{\epsilon_p} + (\boldsymbol{v} + \boldsymbol{\epsilon_v})(t_i - t_0)$ $\check{\boldsymbol{v}}_i = \boldsymbol{v} + \boldsymbol{\epsilon_v}$

$\check{d}_i^2 = |\boldsymbol{v} + \boldsymbol{\epsilon_v}|^2(t_i - t_0)^2 + 2(\boldsymbol{v} + \boldsymbol{\epsilon_v})'(\boldsymbol{p}_0 + \boldsymbol{\epsilon_p})(t_i - t_0) + |\boldsymbol{p}_0 + \boldsymbol{\epsilon_p}|^2$

$\check{\tau}_{i,j} = \frac{2}{c}\sqrt{v^2(t_i - t_0)^2 + 2\boldsymbol{v}'\boldsymbol{p_0}(t_i - t_0) + d_0^2} + t_p + \epsilon_{ti,j}$

$\check{\tau}_{i,j} = \tau_i + \epsilon_{t_p i,j}$, where $\epsilon_{t_p i,j} \sim N(\frac{\rho}{2}, \frac{\rho^2}{12})$ is a uniform distribution in $[0, \rho]$.

$\check{d}_i = d_i + \frac{1}{d_i}(\boldsymbol{p_0} + \boldsymbol{v}(t_i - t_0))(\boldsymbol{\epsilon_p} + \boldsymbol{\epsilon_v}(t_i - t_0))' = d_i + \epsilon_{d_i}$,

where $\epsilon_{d_i} = \frac{1}{d_i}\boldsymbol{p}_i\boldsymbol{\epsilon}_i'$, $\boldsymbol{\epsilon}_i = \boldsymbol{\epsilon_p} + \boldsymbol{\epsilon_v}(t_i - t_0)$, $\boldsymbol{\epsilon_p} \sim N(\mathbf{0}, \boldsymbol{\sigma_p^2})$ and $\boldsymbol{\epsilon_v} \sim N(\mathbf{0}, \boldsymbol{\sigma_v^2})$ are normal distribution.

An estimate of the mean processing time and its standard deviation will create the upper bound used in the first detection method, boundary checking.

Error estimation on $\hat{T_p}$

$\hat{T_p} = \frac{1}{n}\sum_{i=1}^{n}(\hat{\tau_i} - \frac{2}{c}\check{d_i}) = \frac{1}{n}\sum_{i=1}^{n}(\hat{\tau_i} - \frac{2}{c}(d_i + \epsilon_{d_i}))$

$\xi(\hat{T_p}) = \frac{1}{n}\sum_{i=1}^{n}(\xi(\hat{\tau_i}) - \frac{2}{c}\xi(\check{d_i})) = \frac{1}{n}\sum_{i=1}^{n}(\xi(\hat{\tau_i}) - \frac{2\boldsymbol{p}_i\boldsymbol{\epsilon}'_i}{cd_i})$

$E(\xi(\hat{T_p})) = \frac{1}{n}\sum_{i=1}^{n}(\frac{\rho}{2} + 0) = \frac{\rho}{2}$ and $\sigma^2(\xi(\hat{T_p})) = \frac{\rho^2}{12nm} + \frac{4}{n^2c^2}\sum_{i=1}^{n}(\sigma_p^2 + \sigma_v^2(t_i - t_0)^2)$

Estimations of time and distance are also determinable from the message and parameters.

Error estimation on $\hat{\tau_i}$

$\hat{\tau_i} = \frac{1}{m}\sum_{j=1}^{m}\check{\tau}_{i,j} = \frac{1}{m}\sum_{j=1}^{m}(\tau_i + \epsilon_{t_p i,j}) = \tau_i + \frac{1}{m}\sum_{j=1}^{m}\epsilon_{t_p i,j}$

$\xi(\hat{\tau_i}) = \frac{1}{m}\sum_{j=1}^{m}\xi(\check{\tau}_{i,j}) = \frac{1}{m}\sum_{j=1}^{m}\epsilon_{t_p i,j}$.

So, $E(\xi(\hat{\tau_i})) = \frac{1}{m}\sum_{j=1}^{m}E(\epsilon_{t_p i,j}) = \frac{\rho}{2}$ and $\sigma^2(\xi(\hat{\tau_i})) = \frac{1}{m^2}\sum_{j=1}^{m}\sigma^2(\epsilon_{t_p i,j}) = \frac{\rho^2}{12m}$

Error estimation on $\hat{d_i}$

$$
\begin{aligned}
\hat{d_i} &= \frac{c}{2}(\frac{1}{m}\sum_{j=1}^{m}\check{\tau}_{i,j} - \frac{1}{n}\sum_{k=1}^{n}(\hat{\tau_k} - \frac{2}{c}\check{d_k})) \\
&= \frac{c}{2}(\frac{1}{m}\sum_{j=1}^{m}(\tau_i + \epsilon_{t_p i,j}) - \frac{1}{n}\sum_{k=1}^{n}(\tau_k + \frac{1}{m}\sum_{j=1}^{m}\epsilon_{t_p k,j} - \frac{2}{c}(d_k + \epsilon_{d_k}))) \\
&= \frac{c}{2}(\tau_i + \frac{1}{m}\sum_{j=1}^{m}(\epsilon_{t_p i,j}) - \frac{1}{n}\sum_{k=1}^{n}(\frac{1}{m}\sum_{j=1}^{m}\epsilon_{t_p k,j}) - \frac{1}{n}\sum_{k=1}^{n}(\tau_k - \frac{2}{c}(d_k + \epsilon_{d_k}))) \\
&= \frac{c}{2}(\tau_i - t_p + \frac{1}{m}\sum_{j=1}^{m}\epsilon_{t_p i,j} - \frac{1}{nm}\sum_{k=1}^{n}\sum_{j=1}^{m}\epsilon_{t_p k,j} - \frac{1}{n}\sum_{k=1}^{n}\frac{2}{c}\epsilon_{d_k}) \\
&= d_i + \frac{c}{2}(\frac{n-1}{nm}\sum_{j=1}^{m}\epsilon_{t_p i,j} - \frac{1}{nm}\sum_{k=1,k\neq i}^{n}\sum_{j=1}^{m}\epsilon_{t_p k,j} - \frac{1}{n}\sum_{k=1}^{n}\frac{2}{c}\epsilon_{d_k})
\end{aligned}
$$

$\xi(\hat{d_i}) = \frac{c}{2}(\frac{n-1}{nm}\sum_{j=1}^{m}\epsilon_{t_p i,j} - \frac{1}{nm}\sum_{k=1,k\neq i}^{n}\sum_{j=1}^{m}\epsilon_{t_p k,j} - \frac{1}{n}\sum_{k=1}^{n}\frac{2}{c}\epsilon_{d_k})$

$E(\xi(\hat{d_i})) = 0$ and $\sigma^2(\hat{d_i}) = \frac{c^2(n-1)\rho^2}{48nm} + \frac{1}{n^2}\sum_{k=1}^{n}(\sigma_p^2 + \sigma_v^2(t_k - t_0)^2)$

VI. IMPLEMENTATION

SITL

The SITL (software in the loop) simulator facilitates UAV testing without any hardware. Built using ordinary C++, it takes advantage of the fact that ArduPilot code is a portable autopilot that can run on a very wide variety of platforms. A Hardware Abstraction Layer (HAL) code allows this portability. SITL allows for easy modification and testing, so a new Class file was created to support the Distance-Bounding procedure.

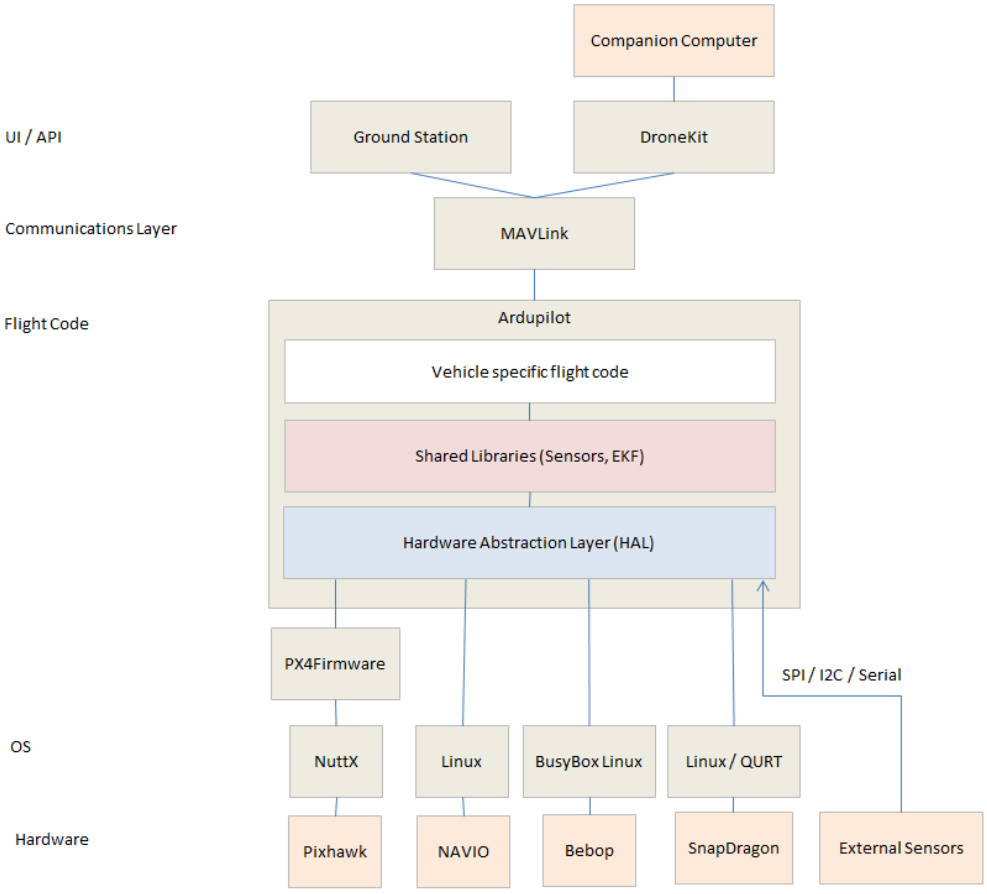

Figure VI.1: ArduPilot Architecture

Ghost Plane

Other aircraft in the simulation are generated in the SIM_ADSB class. The aircraft is generated with an initial position and velocity relative to the simulated copter using a simple cartesian system. When the vehicle finishes its update cycle, the class constructs a MAVLink message of type ADSB_VEHICLE as seen in Figure VI.2. These messages arrive via a MAVLink communication channel to the simulated ArudPilot vehicle. The ghost vehicle information resides in the simulator object (_SITL) and does not have the full range of functionality like a simulated ArduPilot board. They will not attempt to make any collision avoidance maneuvers.

ADSB_VEHICLE (#246)

The location and information of an ADSB vehicle

Field Name	Type	Units	Values	Description
ICAO_address	uint32_t			ICAO address
lat	int32_t	degE7		Latitude
lon	int32_t	degE7		Longitude
altitude_type	uint8_t		ADSB_ALTITUDE_TYPE	ADSB altitude type.
altitude	int32_t	mm		Altitude(ASL)
heading	uint16_t	cdeg		Course over ground
hor_velocity	uint16_t	cm/s		The horizontal velocity
ver_velocity	int16_t	cm/s		The vertical velocity. Positive is up
callsign	char[9]			The callsign, 8+null
emitter_type	uint8_t		ADSB_EMITTER_TYPE	ADSB emitter type.
tslc	uint8_t	s		Time since last communication in seconds
flags	uint16_t		ADSB_FLAGS	Bitmap to indicate various statuses including valid data fields
squawk	uint16_t			Squawk code

Figure VI.2: MAVLink ADS-B Vehicle Message

ADSB in SITL

The AP_ADSB class handles ADS-B processing for the ArduPilot software. It maintains the list of tracked aircraft, handles unpacking of ADS-B messages, and performs ADS-B out functions. The list has an adjustable max limit and max distance for tracking parameter than can be adjusted depending on the UAV hardware restraints. If the list is full, the class will check to see if the new vehicle is closer than the furthest vehicle it is currently tracking. If it is closer, the furthest vehicle will be dropped to make room for the new vehicle.

The update cycle is used to ensure all the vehicles on the list have been reporting regularly. If too much time elapses without a new message, the vehicle will be removed from the list. Since this class handles the ADS-B messages, it is here the call to the distance-bounding process will be made. For testing purposes it will be done every time an ADS-B message is received.

Details

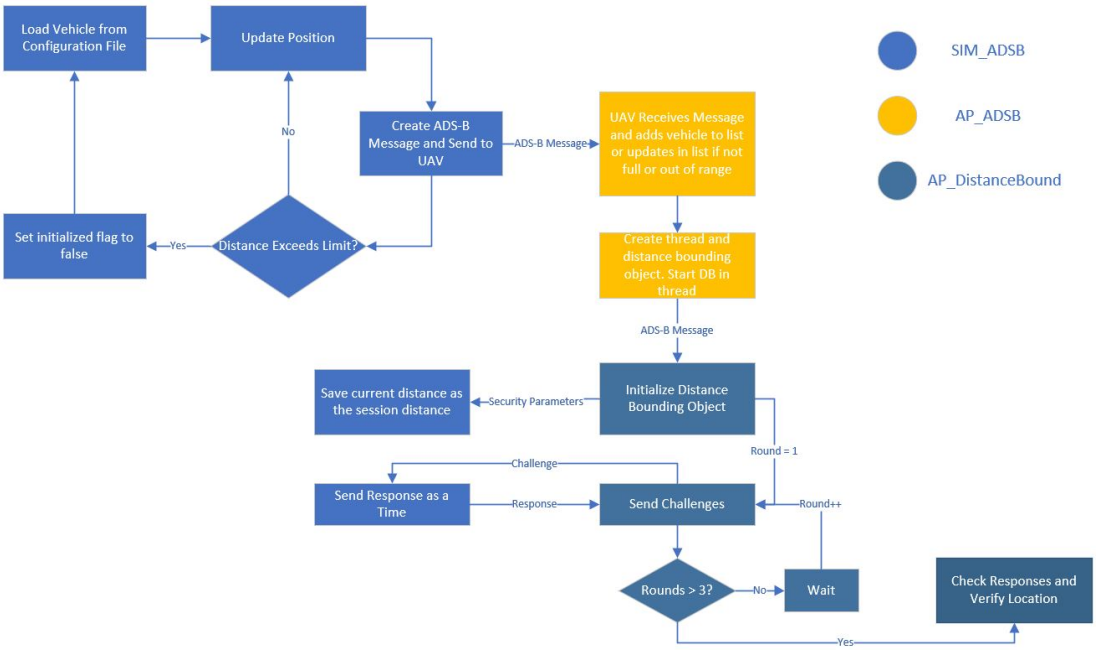

Figure VI.3: Distance-Bounding in SITL Work Flow

Ghost Vehicle Generation

Instead of randomly generating parameters, this function allows the simulator to parse a JSON file for flight configurations. This includes parameters that were not normally included during the ghost aircraft initialization. These parameters need to be visible to the UAV only for the purpose of creating logs and determining error. In practice, the error values and processing time lower bound would be predetermined by the equipment. The wait times between rounds is generated by the UAV in practice, it is only included here for simulation testing.

Anytime a new flight needs to be generated, the index for that flight in the list sets the variable $initialised$ to false. This prompts the next update cycle to load in a configuration. The altitude was kept at zero to ensure both the UAV and the ghost were set for collision without having to hover the UAV. The velocity obtained from the JSON needs to be split into components directed toward the UAV. First we get the angle, knowing the UAV will be origin: $atan(slope)$. Then the components can be extracted using sin and cos functions of the angle multiplied by velocity and setting the sign.

Algorithm 1 Configuring the Ghost Aircraft

```
if !initialised then
    initialised = true
    getNextJsonVehicleConfiguration()
    position.x = Aircraft::randNormal(0, sitl->adsbRadius/4)
    position.y = Aircraft::randNormal(0, sitl->adsbRadius/4)
    position.z = 0
    double angle = atan(((0 - position.y)/(0 - position.x)))
    velocity.x = (velocity * (cos(angle)))
    velocity.y = (velocity * (sin(angle)))
end if
```

The ghost aircraft needs to generate an ADS-B message to send the UAV. The error for those measurements is calculated first. The error for position and velocity is then added to the values before they are converted for an ADS-B Message.
When the ghost aircraft receives the distance-bounding request, there is a setup function to initialize any variables it needs from the UAV. Since the simulation is not using real time-of-arrival measurements between the ghost and UAV, the ghost aircraft instead sends back the T_i based on its location, processing time, and error. This response has two parts, a response bit and a response time. The ghost aircraft will use its current location when it receives the distance-bounding request as the session starting location. The wait times are passed to the ghost aircraft since the session will

not occur in real time. The ghost aircraft can then use the wait times to respond to the UAV with locations approximating if real time had been occurring. There is a small amount of time unaccounted for in the system, that is the time from when the ADS-B message is sent until the distance-bounding request is received. This is expected to generate the real scenario, since ADS-B messages do not contain a time stamp of creation.

Algorithm 2 Ghost Aircraft Response Function

procedure RESPONSE FUNCTION(challengeBit)

if $roundNumber \ \% \ challengesPerRound == 0$ **then**

$tempx \leftarrow velocity.x(roundWaitTime)$

$tempy \leftarrow velocity.y(roundWaitTime)$

$tempz \leftarrow velocity.z(roundWaitTime)$

$sessionDistance+ = \sqrt{tempx^2 + tempy^2 + tempz^2}$

end if

if $challengeBit == 1$ **then**

$response.bit \leftarrow R1[0]$

else

$response.bit \leftarrow R0[0]$

end if

$R0$ »= 1

$R1$ »= 1

$distanceTime \leftarrow \frac{2sessionDistance}{speedOfLight}$

$noise \leftarrow randomNormal(0, errorMax)$

$response.time \leftarrow distanceTime + processTime + noise$

return $response$

end procedure

DistanceBound Class

The constructor for the AP_DistanceBound class performs the following steps:

1. Retrieve SITL object
2. Log time of DistanceBound creation, this should be very near the send time of the ADS-B message
3. Initialize the security parameters
4. Acquire the simulation reference to the vehicle for communication

The values it retrieves from the simulation vehicle is only for simulation purposes. In practice these values are not retrieved from the prover however, they are predetermined such as lower and upper bound of processing time, GPS error ranges, and time delays. Once the Class is created, it is handed its own thread to allow the multiple distance-bounding sessions to occur simultaneously. The thread starts in the slow phase function3, which has many responsibilities before starting the fast phase.

Algorithm 3 Multi-point Distance Bounding Slow Phase

procedure SLOW PHASE(ADS-B Message)

Save Time of Creation

Authenticate Prover to share secret K

Generate Nonce N_v

Exchange Nonces with Prover ▷ Send N_v and Receive N_p

hashFunctionH(K, N_v, N_p) ▷ Generate Bit Sequences $R0$ and $R1$

Generate Delay Times Between Rounds

Start Fast Phase

end procedure

The fast phase, algorithm 4, is a simple procedure. It generates challenges and times the interactions between the UAV and the Ghost aircraft. The hardware abstraction layer provides access to the current simulation time in microseconds.

Algorithm 4 Multi-point Distance Bounding Fast Phase

```
procedure FAST PHASE
    for i ← 0, number of rounds do
        for j ← 0, challenges per round do
            Generate Challenge C_{i,j}
            Send C_{i,j} and Start Timer
            while R_{i,j} is NULL do
                WAIT;                                   ▷ Should not wait indef.
            end while
            Stop Timer and Save R_{i,j}
        end for
        Wait for Dealy_i microseconds
    end for
    Return Bit Sequence
end procedure
```

The verification process will occur in the finalization algorithm 5. This procedure makes sense of the the information gathered during the fast phase. First, the number of failed challenges needs to be removed from further processing. These will not be considered during the analysis of processing time and reduce the average. It does not currently take any action when a message fails the boundary check or incorrect response threshold. This functionality should be decided by the user to decide. If the standard deviations is set low, the user may want to interrogate these flagged aircraft further before removing them. Calculations for processing time error and deviation can be found Chapter V.

Algorithm 5 Finalize Function in AP DistanceBound Class

procedure FINALIZE

for $i \leftarrow 0, totalRounds$ **do** ▷ Predict future locations from ADS-B message

$EstimatedDistance_i \leftarrow ClaimedDistance + Delay_i(ClaimedVelocity)$

end for

for $i \leftarrow 0$, Total Challenges **do** ▷ Check correctness and adding values

Correct Answer $\leftarrow R0_i$ or $R1_i$ based on $Challenge_i$

Travel Time $\leftarrow \frac{2(EstimatedDistance_{round})}{c}$

if Correct Answer == $responses_i$ **then**

Add the time for challenge to this round

Add the time for challenge minus travel time to processing time

else

Increment and Track Wrong Answer Count

end if

end for

if Wrong Answers < Threshold **then**

Average Response Times for Each Round and Processing Time for Session

Calculate Estimated Mean Processing Time and Standard Deviation

Upperbound = Estimated Mean Processing Time + Deviations(Std Dev)

Lowerbound = Estimated Mean Processing Time - Deviations(Std Dev)

if Lowerbound < Average Processing time < Upperbound **then**

Verified by Successful Distance Bounding

else

Failed Distance Bounding Boundary Check

end if

else

Failed Answer Threshold

end if

end procedure

VII. EXPERIMENTS

Settings

An overview of the settings and their ranges can be found in VII.1. The sections below discuss this settings in more detail, including what they impact and why these ranges were chosen. The configurations by flight number can be found in Appendix A.

Table VII.1: Simulation Setting Variables

Variable	**Range**
Velocity(m/s)	5,10,15,20,30,40
Distance(m)	0 to 500
Wait Time(us)	1000,100000,250000,500000
Rounds/Challenges	16/3
Processing Time Lower Bound (us)	3,30,300
Noise	$\frac{10}{c}, \frac{50}{c}, \frac{100}{c}, \frac{300}{c}, \frac{500}{c}$
Attack	True, False

Ghost UAV Settings

Velocity was the first consideration that could have significant impact on the success of distance bounding. The experiment should test for the spectrum of UAV operating speeds. The FAA dictates no UAV should exceed 100mph(87 knots) [11], which is approximately 45m/s. Using that information, velocity was given an upper bound of 40m/s and a lower bound of 5 m/s.

Distance has several considerations in the experiment. The distance between a verifier and prover may have impact on the results of the distance-bounding. It also needs to be significant enough for the verifier to make decisions based on the information received. A distance resulting in time segments smaller than the processing time will be indistinguishable from a small deviation on the processing time itself. In that scenario, it is also unlikely the verifier will be able to take an evasive action before collision

could occur. UAVs have significantly less operating range than standard or commercial aircraft, so the experiment was designed to stop tracking or maintaining any simulated aircraft past a 500 meter radius of the verifier.

Angle of Approach as mentioned in the threat model, this is important to the scope of this research. In all scenarios, the prover UAV flight plan will include a collision with the static verifier, origin on a Cartesian coordinate system. The formula and method of determining this is shown in Chapter VI. The angle is calculated dynamically after a position is randomly generated within the acceptable range.

Distance-bounding Settings

Rounds and Challenges are one of the major contributions of this work. Rounds are a collection of challenges performed together at an interval based on the same ADS-B message. Challenges are the number of bit exchanges to occur that round. Both the verifier and the prover will know these parameters prior to distance-bounding. These variables should be fine-tuned by the experiment to maximize reduction of noise and error at the lowest cost of overhead. The protocol needs to be able to successfully detect ghost aircraft with 99.7 percent reliability.

Wait Time is another distance-bounding parameter unique to this work. The wait time is a randomly determined pause between rounds in a distance-bounding session. This parameter is believed to have two impacts on the protocol. First, it will provide additional security, as the prover does not know the wait time between rounds. Pre-send attacks are less likely to succeed with this added security feature. The second impact it will have is on the success of the distance-bounding session. The change in distance during the session will largely be dependent upon this variable and the UAV speed. If there is not a detectable change in position or distance, the effectiveness of the protocol would be significantly reduced. Multiple sessions would provide no improvement for two mathematically static objects.

Processing Time of the distance-bounding device is a crucial variable to the estimation equations. When designing the experiments, many of the security parameters are not

considered.

Noise represents the amount of variation in the processing time. Values were determined as the amount of distance error at the speed of light, $\frac{distance}{c}$.

Simulation Results

The simulation is run on two identical configuration files, switching the attack flag to true during the second iteration. Each iteration is 360 unique configurations, changing one variable. The UAV in the simulator will perform a distance-bounding session for every ADS-B message it receives. This means more sessions are conducted during the slower flights. The simulator creates log files for each flight containing JSON data for every session during that flight. The log files are analyzed in R to produce all the following graphics, tables, and explanations. It finds the mean and standard deviation using the $mean()$ and $sd()$ functions in R on the data received from the simulation to make those calculations. This is done on raw error data files created by extracting the information from the JSON files.

The two categories of error to analyze are distance and processing time. Since the algorithm assumes the claimed distance is real, any distance error is the a reflection of the time between generating the message and receiving the distance bounding request. Any difference between the average time for the round and estimations will indicate a related difference between the claimed position or velocity and the actual values. The detection process will accumulate this time discrepancy into the processing time calculation.

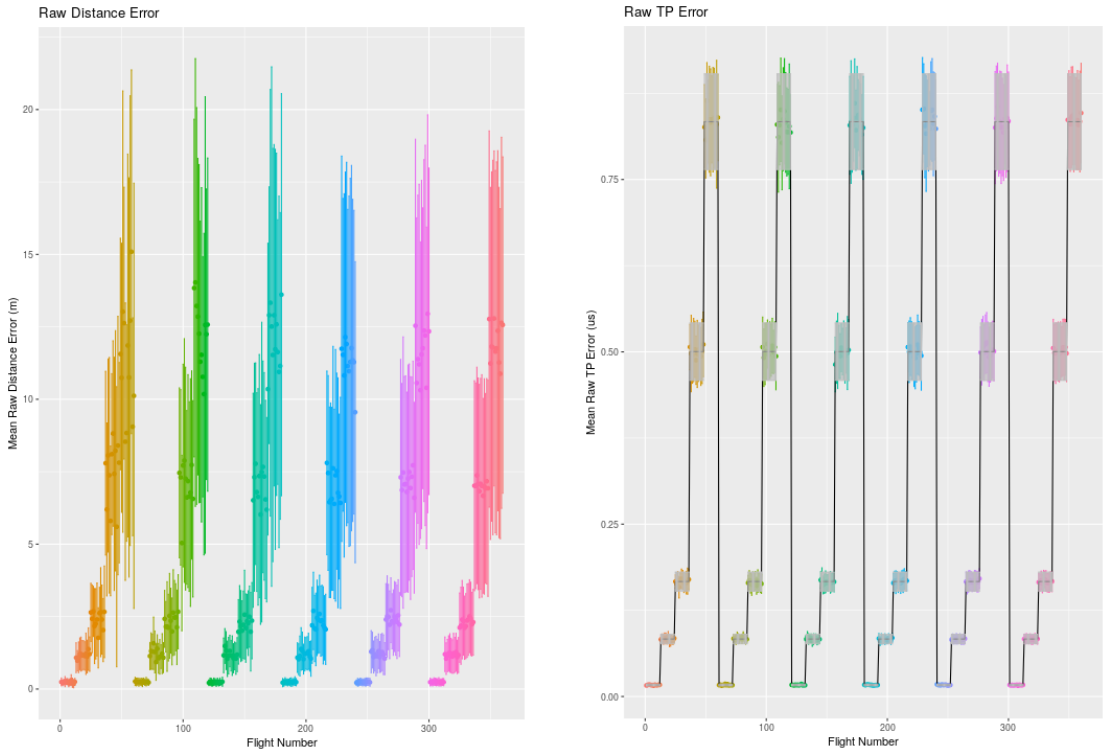

Figure VII.1: Distance and Processing Time Error Legitimate: All Flights

The first iteration through the simulation is legitimate flights. None of the flights are broadcasting fraudulent messages. The distance error increases as the processing time noise increases for each group. The larger appearing groups are velocity. Although it appears distance error may be slightly decreasing as velocity decreases, it is only a smaller deviation from a larger sample set. The speed is slow enough to allow more sessions to occur during that configuration.

The processing time graph in VII.1 uses a black line to illustrate the mean processing time with gray error bars set to standard deviation off the mean. Legitimate messages are compared to the estimates with very accurate results. Much like distance, the higher numbered flights have more samples and rarely exceed the boundaries. A breakdown of the first 60 flights can be seen in VII.3, as the data sets are velocity groupings of 60 flights. All 60 have the same velocity, but the other settings will vary by flight. This figure illustrates the predictability of the distance-bounding with ADS-B when measuring a legitimate prover. The changing of settings show consistent results throughout the experiment.

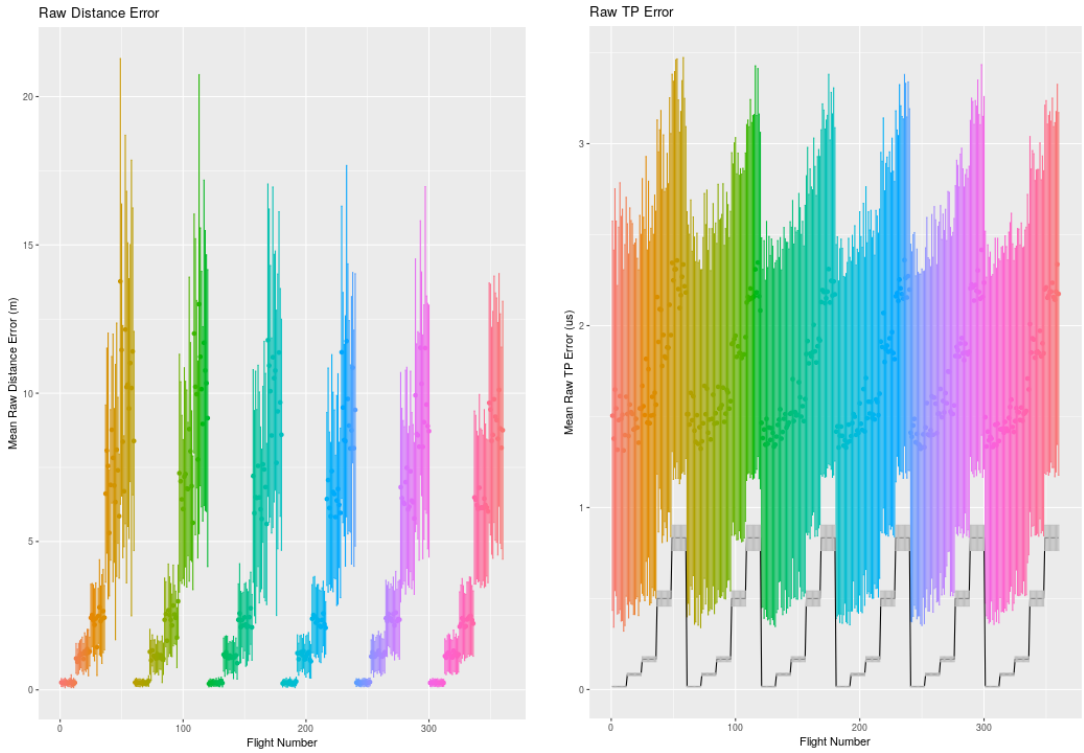

Figure VII.2: Distance and Processing Time Error Fraudulent: All Flights

The second set of data tested, shown in VII.2 called attack data, shows the same calculations and comparisons from the legitimate set. The flight configurations are identical with the only change being the attack flag is set to true. The distance error shows an identical resemblance to the legitimate flights. This is as expected due to the algorithms assumption the claimed distance in the ADS-B message is accurate. Assuming this distance is accurate will make any distance calculations irrelevant, putting weight on the processing time calculations instead.

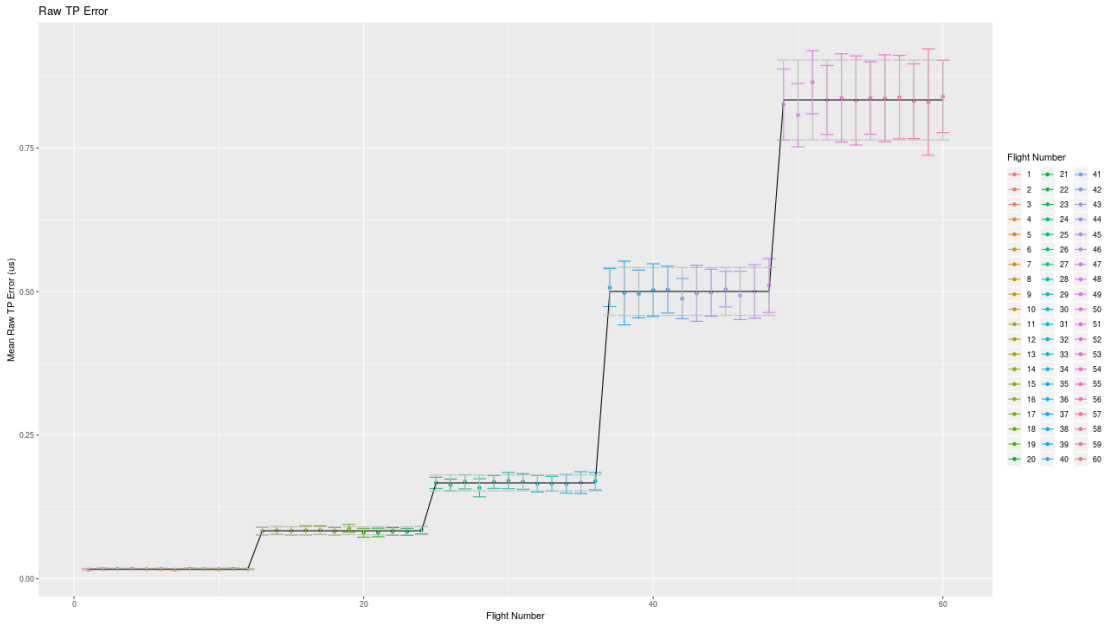

Figure VII.3: Processing Time Error Legitimate: Configurations 1 to 60

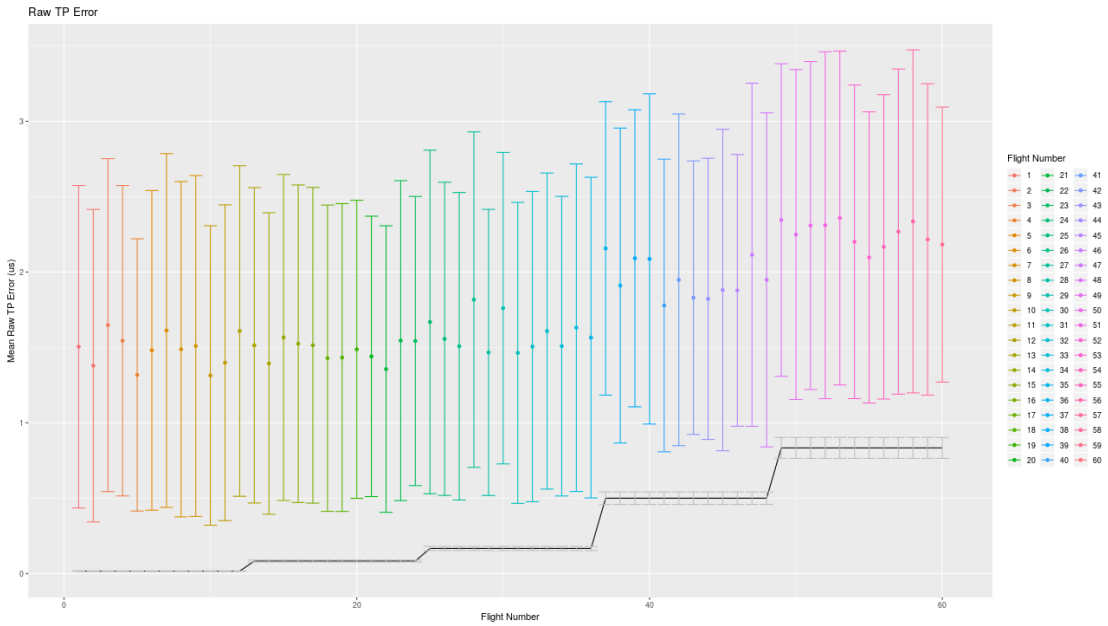

Figure VII.4: Processing Time Error Fraudulent: Configurations 1 to 60

As with the legitimate data, the black line represents the mean and grey error bars represent standard deviation from that mean. The flights are the same configuration, so the estimates shown in VII.2 are the same as the ones in VII.1. Here we can see the processing time is always outside the boundaries set by the standard deviation. A closer look into one velocity grouping can be seen in VII.4. Like the legitimate set, velocity remains the same while other settings change. The processing time noise determines

the range of distance an attacker can use. As the noise increases, the calculation deviation gets closer to overlapping the estimated deviation signifying false negatives. Comparing figures VII.4 and VII.3 illuminates the glaring differences between an attack and legitimate broadcasts. The mean line will be identical in both figures.

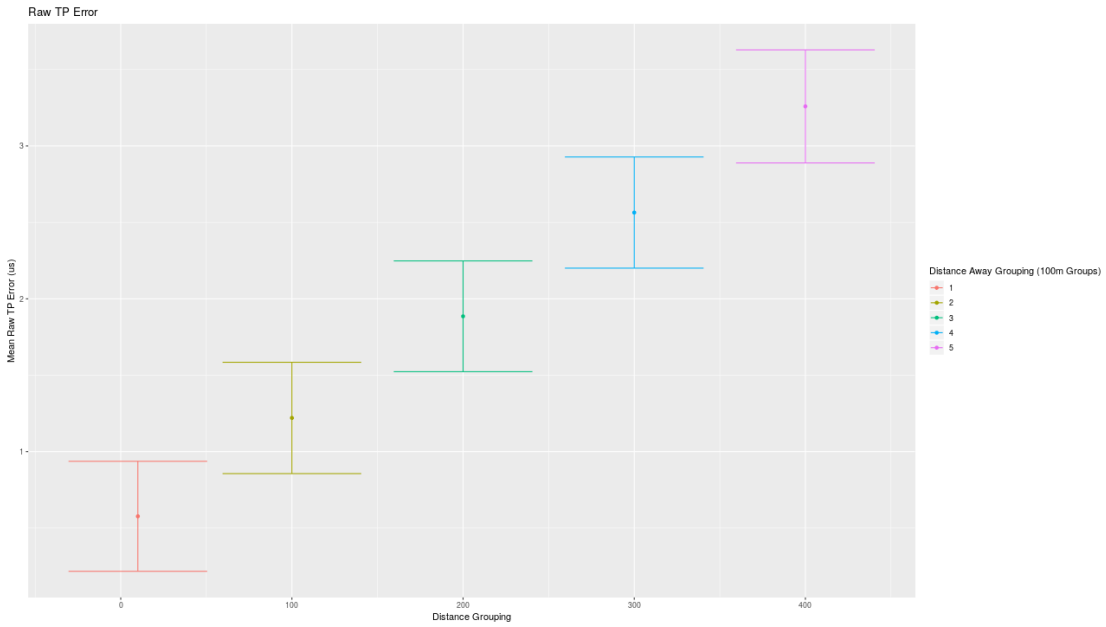

Figure VII.5: Attack Processing Time Error by Distance

The deviation appears to be much larger for the attack data. This is directly related to the difference between the real and claimed positions, shown in VII.5. Since the flights can start anywhere up to 500 meters away, the attack flag changes any flight over 10 meters to the coordinate (7,7). This is approximately 10 meters away from the UAV. If the flight was generated 500 meters away, this error will be significantly larger than if the flight had been 15 meters away.

Overhead

The distance-bounding process will add some additional computation and communication overhead to the UAV. Distance-bounding processes should be conducted over communication channel reserved for it to prevent interference with other communication processes with a ground station or controller.

Errors

Table VII.2: Fraudulent Flight Detection

Group	Total Sessions	Failed Flights	Detection Error
Attackers	17,952	17,301	3.626%
Legitimate	18,441	23	0.125%

No sessions when the prover was within 10 meters of the verifier were considered in the calculation. Distances ranged from 10 meters to 500 meters from the verifier. The boundary was set to three standard deviations above the mean, using a normal distribution for noise. Both groups contained 360 flights with identical configurations in each group. The error in detection is shown in VII.2 and a breakdown by distance in VII.3

Less than half of a percent of legitimate messages are flagged as suspicious, the details of which flights and sessions found in Appendix A. These primarily occurred when the distance exceeded 100 meters, but were very small in quantity to specifically contribute this to the distance as seen in VII.5. Using a boundary line of three standard deviations, we expected up to 0.3% of data to fall outside the boundary since the noise is introduced as a normal distribution.

Just over 3.6% of fraudulent messages were not flagged by the algorithm as shown in VII.4. Over 99.98% of these occurred when the broadcaster was within 50 meters of the UAV. The broadcaster always claimed to be closer than they were actually located as indicated in the threat model. When the processing time is large and the distance is small, the false negative rate will increase due to the noise exceeding the signal propagation time. The time for signal to travel 50 meters back and forth is $\frac{2distance}{c}$ or $3.336E^{-7}$, while noise testing settings maxed at $\frac{500}{c}$ or $1.669E^{-6}$. This is another indication that the processing time needs to small and consistent.

Table VII.3: Detection Error Histogram Tables

Table VII.4: Undetected Fraudulent Messages by Distances

Bin Limit(m)	Count
15	189
20	127
30	186
40	100
50	40
60	7
70	1
80	1
90	0
100+	0
Total	651

Table VII.5: Failed Legitimate Messages by Distances

Bin Limit(m)	Count
15	0
20	0
30	2
40	0
50	1
60	0
70	0
80	2
90	2
100	0
200	8
300	2
400	4
500	2
Total	23

Security

If an attacker attempts to purely guess all the challenges during the fast phase, the probability an attack would succeed is $(\frac{1}{2})^{nm}$. The use of void challenges can improve this security.

Another set of attacks on distance-bounding protocols are pre-ask or post-ask strategy and Early-reply attacks. Pre-ask strategy is a form of mafia fraud in which the attack relays the initial slow phase between the verifier and the prover, then starts the fast phase with the prover before the verifier starts it. The post-ask strategy is very similar in that it relays the initial slow phase, but then it conducts the fast phase with the verifier alone. After the fast phase, the attacker communicates with the prover again now with the correct challenges. This type of attack is effective when the protocol contains a second slow phase. In early-reply, the dishonest prover sends response $R_i'^{C_i}$ in anticipation before receiving the challenge C_i.

The attacker has an increase to $(\frac{3}{4})^{nm}$ probability of success in a pre-ask mafia attack as well as an early-reply distance fraud attack.

VIII. CONCLUSION

Conclusion

The ADS-B protocol is mandated for use in manned aircraft and becoming increasingly popular in UAVs. The safety benefits introduced by this system are limited by the lack of security features to defend it. Further, any systems reliant upon it are also at risk and unappealing to implement without first securing the source of information. Authentication fails to consider a compromised device, while location verification provides a more robust solution for collision avoidance problem.

An attacker could exploit UAVs using ADS-B based collision avoidance, forcing them to make avoidance maneuvers and alter the flight path. ADS-B based collision avoidance can be secured, however, using a new multi-point distance-bounding protocol. The process uses multiple points along a provers flight path predicted by the verifier from the information obtained in ADS-B. A randomly generated delay between these rounds adds to the security of the protocol. The protocol lacks a final signature phase to improve the speed of the protocol and reduce computational overhead. If the prover passes the initial threshold of successful challenges, i.e. correct responses to verifier, the processing time is used to analyze session responses for fraud.

The averaging process over the session allows the protocol to reduce errors and noise to trivial amounts in legitimate sessions. An attacker made fraudulent claims purporting a smaller distance from the verifier, it would need to take advantage of the processing time fluctuation in combination with early-reply or other distance fraud strategies. This new protocol detects this fraud and identifies it to investigate further. The threshold for detecting boundary violations can be adjusted and tuned to the situation. Beyond boundary detection, further analysis of the processing time across sessions or even within the same session can show erratic or irregular behavior. An oscillating or increasing/decreasing processing time across the session may indicate fraudulent behavior from the proving agent.

Testing the protocol in a SITL environment was successful in identifying 96.374% of the fraudulent distance-bounding sessions with over 99.98% of the failed sessions occurring within 50 meters of the verifier. Consistent and small processing times could further reduce the false negative rating even more. False positive measurements only occurred on 0.125% of the legitimate sessions, well within the expected 0.3% prediction due to the distribution of processing time noise.

Future Work

This research has several branches for testing and improvement. Currently, the implementation is only functional for the SITL simulations. It has only been tested in this capacity. Hardware in the loop (HITL) simulations would be the next step before live flight tests.

A distance-bounding device capable of meeting the requirements for a UAV needs to be designed. Current candidates for this are small devices such as Arduino or RaspberryPi. More testing needs to be conducted on identifying and maintaining consistent processing times. Any improvement to the processing time estimations would significantly strengthen the new distance-bounding protocol.

Future work on the protocol can take on many forms. Data fusion is a good candidate for improving detection confidence. Concepts from [30] include a set of plausibility checks. After conducting the distance-bounding, a series of checks are performed for tampering:

1. A vehicle cannot claim to be located further away than maximum transmission range.

2. A vehicle speed cannot exceed the mechanical limit to move further than possible between messages.

3. A vehicle cannot claim to be located off the roadway (In aircraft, the system may check to see if terrain conflicts with the position)

APPENDIX SECTION

APPENDIX A

Flight Configurations 1 to 60

ICAO	**Vel.** $\frac{m}{s}$	**Max Error**	**tp**	**Wait Times**		**ICAO**	**Vel.** $\frac{m}{s}$	**Max Error**	**tp**	**Wait Times**
1	40	10/c	3	500000		31	40	100/c	30	100000
2	40	10/c	3	250000		32	40	100/c	30	10000
3	40	10/c	3	100000		33	40	100/c	300	500000
4	40	10/c	3	10000		34	40	100/c	300	250000
5	40	10/c	30	500000		35	40	100/c	300	100000
6	40	10/c	30	250000		36	40	100/c	300	10000
7	40	10/c	30	100000		37	40	300/c	3	500000
8	40	10/c	30	10000		38	40	300/c	3	250000
9	40	10/c	300	500000		39	40	300/c	3	100000
10	40	10/c	300	250000		40	40	300/c	3	10000
11	40	10/c	300	100000		41	40	300/c	30	500000
12	40	10/c	300	10000		42	40	300/c	30	250000
13	40	50/c	3	500000		43	40	300/c	30	100000
14	40	50/c	3	250000		44	40	300/c	30	10000
15	40	50/c	3	100000		45	40	300/c	300	500000
16	40	50/c	3	10000		46	40	300/c	300	250000
17	40	50/c	30	500000		47	40	300/c	300	100000
18	40	50/c	30	250000		48	40	300/c	300	10000
19	40	50/c	30	100000		49	40	500/c	3	500000
20	40	50/c	30	10000		50	40	500/c	3	250000
21	40	50/c	300	500000		51	40	500/c	3	100000
22	40	50/c	300	250000		52	40	500/c	3	10000
23	40	50/c	300	100000		53	40	500/c	30	500000
24	40	50/c	300	10000		54	40	500/c	30	250000
25	40	100/c	3	500000		55	40	500/c	30	100000
26	40	100/c	3	250000		56	40	500/c	30	10000
27	40	100/c	3	100000		57	40	500/c	300	500000
28	40	100/c	3	10000		58	40	500/c	300	250000
29	40	100/c	30	500000		59	40	500/c	300	100000
30	40	100/c	30	250000		60	40	500/c	300	10000

Flight Configurations 61 to 120

ICAO	**Vel.** $\frac{m}{s}$	**Max Error**	**tp**	**Wait Times**		**ICAO**	**Vel.** $\frac{m}{s}$	**Max Error**	**tp**	**Wait Times**
61	30	10/c	3	500000		91	30	100/c	30	100000
62	30	10/c	3	250000		92	30	100/c	30	10000
63	30	10/c	3	100000		93	30	100/c	300	500000
64	30	10/c	3	10000		94	30	100/c	300	250000
65	30	10/c	30	500000		95	30	100/c	300	100000
66	30	10/c	30	250000		96	30	100/c	300	10000
67	30	10/c	30	100000		97	30	300/c	3	500000
68	30	10/c	30	10000		98	30	300/c	3	250000
69	30	10/c	300	500000		99	30	300/c	3	100000
70	30	10/c	300	250000		100	30	300/c	3	10000
71	30	10/c	300	100000		101	30	300/c	30	500000
72	30	10/c	300	10000		102	30	300/c	30	250000
73	30	50/c	3	500000		103	30	300/c	30	100000
74	30	50/c	3	250000		104	30	300/c	30	10000
75	30	50/c	3	100000		105	30	300/c	300	500000
76	30	50/c	3	10000		106	30	300/c	300	250000
77	30	50/c	30	500000		107	30	300/c	300	100000
78	30	50/c	30	250000		108	30	300/c	300	10000
79	30	50/c	30	100000		109	30	500/c	3	500000
80	30	50/c	30	10000		110	30	500/c	3	250000
81	30	50/c	300	500000		111	30	500/c	3	100000
82	30	50/c	300	250000		112	30	500/c	3	10000
83	30	50/c	300	100000		113	30	500/c	30	500000
84	30	50/c	300	10000		114	30	500/c	30	250000
85	30	100/c	3	500000		115	30	500/c	30	100000
86	30	100/c	3	250000		116	30	500/c	30	10000
87	30	100/c	3	100000		117	30	500/c	300	500000
88	30	100/c	3	10000		118	30	500/c	300	250000
89	30	100/c	30	500000		119	30	500/c	300	100000
90	30	100/c	30	250000		120	30	500/c	300	10000

Flight Configurations 121 to 180

ICAO	Vel.$\frac{m}{s}$	Max Error	tp	Wait Times		ICAO	Vel.$\frac{m}{s}$	Max Error	tp	Wait Times
121	20	10/c	3	500000		151	20	100/c	30	100000
122	20	10/c	3	250000		152	20	100/c	30	10000
123	20	10/c	3	100000		153	20	100/c	300	500000
124	20	10/c	3	10000		154	20	100/c	300	250000
125	20	10/c	30	500000		155	20	100/c	300	100000
126	20	10/c	30	250000		156	20	100/c	300	10000
127	20	10/c	30	100000		157	20	300/c	3	500000
128	20	10/c	30	10000		158	20	300/c	3	250000
129	20	10/c	300	500000		159	20	300/c	3	100000
130	20	10/c	300	250000		160	20	300/c	3	10000
131	20	10/c	300	100000		161	20	300/c	30	500000
132	20	10/c	300	10000		162	20	300/c	30	250000
133	20	50/c	3	500000		163	20	300/c	30	100000
134	20	50/c	3	250000		164	20	300/c	30	10000
135	20	50/c	3	100000		165	20	300/c	300	500000
136	20	50/c	3	10000		166	20	300/c	300	250000
137	20	50/c	30	500000		167	20	300/c	300	100000
138	20	50/c	30	250000		168	20	300/c	300	10000
139	20	50/c	30	100000		169	20	500/c	3	500000
140	20	50/c	30	10000		170	20	500/c	3	250000
141	20	50/c	300	500000		171	20	500/c	3	100000
142	20	50/c	300	250000		172	20	500/c	3	10000
143	20	50/c	300	100000		173	20	500/c	30	500000
144	20	50/c	300	10000		174	20	500/c	30	250000
145	20	100/c	3	500000		175	20	500/c	30	100000
146	20	100/c	3	250000		176	20	500/c	30	10000
147	20	100/c	3	100000		177	20	500/c	300	500000
148	20	100/c	3	10000		178	20	500/c	300	250000
149	20	100/c	30	500000		179	20	500/c	300	100000
150	20	100/c	30	250000		180	20	500/c	300	10000

Flight Configurations 181 to 240

ICAO	**Vel.** $\frac{m}{s}$	**Max Error**	**tp**	**Wait Times**		**ICAO**	**Vel.** $\frac{m}{s}$	**Max Error**	**tp**	**Wait Times**
181	15	10/c	3	500000		211	15	100/c	30	100000
182	15	10/c	3	250000		212	15	100/c	30	10000
183	15	10/c	3	100000		213	15	100/c	300	500000
184	15	10/c	3	10000		214	15	100/c	300	250000
185	15	10/c	30	500000		215	15	100/c	300	100000
186	15	10/c	30	250000		216	15	100/c	300	10000
187	15	10/c	30	100000		217	15	300/c	3	500000
188	15	10/c	30	10000		218	15	300/c	3	250000
189	15	10/c	300	500000		219	15	300/c	3	100000
190	15	10/c	300	250000		220	15	300/c	3	10000
191	15	10/c	300	100000		221	15	300/c	30	500000
192	15	10/c	300	10000		222	15	300/c	30	250000
193	15	50/c	3	500000		223	15	300/c	30	100000
194	15	50/c	3	250000		224	15	300/c	30	10000
195	15	50/c	3	100000		225	15	300/c	300	500000
196	15	50/c	3	10000		226	15	300/c	300	250000
197	15	50/c	30	500000		227	15	300/c	300	100000
198	15	50/c	30	250000		228	15	300/c	300	10000
199	15	50/c	30	100000		229	15	500/c	3	500000
200	15	50/c	30	10000		230	15	500/c	3	250000
201	15	50/c	300	500000		231	15	500/c	3	100000
202	15	50/c	300	250000		232	15	500/c	3	10000
203	15	50/c	300	100000		233	15	500/c	30	500000
204	15	50/c	300	10000		234	15	500/c	30	250000
205	15	100/c	3	500000		235	15	500/c	30	100000
206	15	100/c	3	250000		236	15	500/c	30	10000
207	15	100/c	3	100000		237	15	500/c	300	500000
208	15	100/c	3	10000		238	15	500/c	300	250000
209	15	100/c	30	500000		239	15	500/c	300	100000
210	15	100/c	30	250000		240	15	500/c	300	10000

Flight Configurations 241 to 300

ICAO	**Vel.** $\frac{m}{s}$	**Max Error**	**tp**	**Wait Times**		**ICAO**	**Vel.** $\frac{m}{s}$	**Max Error**	**tp**	**Wait Times**
241	10	10/c	3	500000		271	10	100/c	30	100000
242	10	10/c	3	250000		272	10	100/c	30	10000
243	10	10/c	3	100000		273	10	100/c	300	500000
244	10	10/c	3	10000		274	10	100/c	300	250000
245	10	10/c	30	500000		275	10	100/c	300	100000
246	10	10/c	30	250000		276	10	100/c	300	10000
247	10	10/c	30	100000		277	10	300/c	3	500000
248	10	10/c	30	10000		278	10	300/c	3	250000
249	10	10/c	300	500000		279	10	300/c	3	100000
250	10	10/c	300	250000		280	10	300/c	3	10000
251	10	10/c	300	100000		281	10	300/c	30	500000
252	10	10/c	300	10000		282	10	300/c	30	250000
253	10	50/c	3	500000		283	10	300/c	30	100000
254	10	50/c	3	250000		284	10	300/c	30	10000
255	10	50/c	3	100000		285	10	300/c	300	500000
256	10	50/c	3	10000		286	10	300/c	300	250000
257	10	50/c	30	500000		287	10	300/c	300	100000
258	10	50/c	30	250000		288	10	300/c	300	10000
259	10	50/c	30	100000		289	10	500/c	3	500000
260	10	50/c	30	10000		290	10	500/c	3	250000
261	10	50/c	300	500000		291	10	500/c	3	100000
262	10	50/c	300	250000		292	10	500/c	3	10000
263	10	50/c	300	100000		293	10	500/c	30	500000
264	10	50/c	300	10000		294	10	500/c	30	250000
265	10	100/c	3	500000		295	10	500/c	30	100000
266	10	100/c	3	250000		296	10	500/c	30	10000
267	10	100/c	3	100000		297	10	500/c	300	500000
268	10	100/c	3	10000		298	10	500/c	300	250000
269	10	100/c	30	500000		299	10	500/c	300	100000
270	10	100/c	30	250000		300	10	500/c	300	10000

Flight Configurations 301 to 360

ICAO	**Vel.** $\frac{m}{s}$	**Max Error**	**tp**	**Wait Times**		**ICAO**	**Vel.** $\frac{m}{s}$	**Max Error**	**tp**	**Wait Times**
301	5	10/c	3	500000		331	5	100/c	30	100000
302	5	10/c	3	250000		332	5	100/c	30	10000
303	5	10/c	3	100000		333	5	100/c	300	500000
304	5	10/c	3	10000		334	5	100/c	300	250000
305	5	10/c	30	500000		335	5	100/c	300	100000
306	5	10/c	30	250000		336	5	100/c	300	10000
307	5	10/c	30	100000		337	5	300/c	3	500000
308	5	10/c	30	10000		338	5	300/c	3	250000
309	5	10/c	300	500000		339	5	300/c	3	100000
310	5	10/c	300	250000		340	5	300/c	3	10000
311	5	10/c	300	100000		341	5	300/c	30	500000
312	5	10/c	300	10000		342	5	300/c	30	250000
313	5	50/c	3	500000		343	5	300/c	30	100000
314	5	50/c	3	250000		344	5	300/c	30	10000
315	5	50/c	3	100000		345	5	300/c	300	500000
316	5	50/c	3	10000		346	5	300/c	300	250000
317	5	50/c	30	500000		347	5	300/c	300	100000
318	5	50/c	30	250000		348	5	300/c	300	10000
319	5	50/c	30	100000		349	5	500/c	3	500000
320	5	50/c	30	10000		350	5	500/c	3	250000
321	5	50/c	300	500000		351	5	500/c	3	100000
322	5	50/c	300	250000		352	5	500/c	3	10000
323	5	50/c	300	100000		353	5	500/c	30	500000
324	5	50/c	300	10000		354	5	500/c	30	250000
325	5	100/c	3	500000		355	5	500/c	30	100000
326	5	100/c	3	250000		356	5	500/c	30	10000
327	5	100/c	3	100000		357	5	500/c	300	500000
328	5	100/c	3	10000		358	5	500/c	300	250000
329	5	100/c	30	500000		359	5	500/c	300	100000
330	5	100/c	30	250000		360	5	500/c	300	10000

Flagged Legitimate Flights

ICAO	Session	Distance		ICAO	Session	Distance
2	8	295.08		307	54	40.21
129	6	25.23		309	78	146.79
129	13	165.28		314	27	107.23
182	1	73.03		320	148	460.85
242	9	84.83		324	119	379.95
245	41	193.14		326	50	134.78
247	71	395.39		331	56	123.45
250	6	103.95		331	85	268.42
250	8	83.94		331	124	463.39
262	13	164.07		348	126	300.85
282	50	370.93		360	37	25.14
303	58	77.48				

BIBLIOGRAPHY

[1] A. Abu-Mahfouz and G. P. Hancke. Distance bounding: A practical security solution for real-time location systems. *IEEE Transactions on Industrial Informatics*, 9(1):16–27, Feb 2013.

[2] Ardupilot. ADS-B Receiver. http://ardupilot.org/copter/docs/common-ads-b-receiver.html, 2019.

[3] Gildas Avoine, Muhammed Ali Bingöl, Ioana Boureanu, Srdjan č apkun, Gerhard Hancke, Süleyman Kardaş, Chong Hee Kim, Cédric Lauradoux, Benjamin Martin, Jorge Munilla, Alberto Peinado, Kasper Bonne Rasmussen, Dave Singelée, Aslan Tchamkerten, Rolando Trujillo-Rasua, and Serge Vaudenay. Security of distance-bounding: A survey. *ACM Comput. Surv.*, 51(5):94:1–94:33, September 2018.

[4] Gildas Avoine and Aslan Tchamkerten. An efficient distance bounding rfid authentication protocol: Balancing false-acceptance rate and memory requirement. In *Proceedings of the 12th International Conference on Information Security*, ISC '09, pages 250–261, Berlin, Heidelberg, 2009. Springer-Verlag.

[5] S. Brands and D. Chaum. Distance-bounding protocols. *Advances in Cryptology - EUROCRYPT'93*, 765:344–359, 1993.

[6] Eric Chan-Tin, Victor Heorhiadi, Nicholas Hopper, and Yongdae Kim. The frog-boiling attack: Limitations of secure network coordinate systems. *ACM Trans. Inf. Syst. Secur.*, 14(3):27:1–27:23, November 2011.

[7] Andrei Costin and AurÂŽelien Francillon. Ghost in the Air(Traffic): On insecurity of ADSB protocol and practical attacks on ADS-B devices. Technical report, EURECOM, 2012.

[8] C. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun. Distance hijacking attacks on distance bounding protocols. In *2012 IEEE Symposium on Security and Privacy*, pages 113–127, May 2012.

[9] R. Entezari, H. Bahramgiri, and M. Tajamolian. A mafia and distance fraud high-resistance rfid distance bounding protocol. In *2014 11th International ISC Conference on Information Security and Cryptology*, pages 67–72, Sep. 2014.

[10] FAA. Equip ADS-B. http://www.faa.gov/nextgen/equipadsb, 2018.

[11] FAA. Fact Sheet - Small Unmanned Aircraft Regulations (Part 107). https://www.faa.gov/news/fact_sheets/news_story.cfm?newsId=22615, 2018.

[12] V. Fox, J. Hightower, D. Schulz, and G. Borriello. Bayesian filtering for location estimation. *IEEE Pervasive Computing*, 2(3):24–33, July 2003.

[13] G. P. Hancke. Distance-bounding for rfid: Effectiveness of 'terrorist fraud' in the presence of bit errors. In *2012 IEEE International Conference on RFID-Technologies and Applications (RFID-TA)*, pages 91–96, Nov 2012.

[14] G. P. Hancke and M. G. Kuhn. An rfid distance bounding protocol. In *First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05)*, pages 67–73, Sep. 2005.

[15] D. Jeon, Y. Eun, and H. Kim. Estimation fusion with radar and ads-b for air traffic surveillance. In *Proceedings of the 16th International Conference on Information Fusion*, pages 1328–1335, July 2013.

[16] C. H. Kim. Security analysis of ykhl distance bounding protocol with adjustable false acceptance rate. *IEEE Communications Letters*, 15(10):1078–1080, October 2011.

[17] Chong Hee Kim, Gildas Avoine, François Koeune, François-Xavier Standaert, and Olivier Pereira. The swiss-knife rfid distance bounding protocol. In Pil Joong Lee and Jung Hee Cheon, editors, *Information Security and Cryptology – ICISC 2008*, pages 98–115, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[18] Y. Kim and S. Kim. Rfid distance bounding protocol using m-ary challenges. In *ICTC 2011*, pages 782–783, Sep. 2011.

[19] S. Kleber, R. W. van der Heijden, H. Kopp, and F. Kargl. Terrorist fraud resistance of distance bounding protocols employing physical unclonable functions. In *2015 International Conference and Workshops on Networked Systems (NetSys)*, pages 1–8, March 2015.

[20] S. Lee, J. S. Kim, S. J. Hong, and J. Kim. Distance bounding with delayed responses. *IEEE Communications Letters*, 16(9):1478–1481, Sep. 2012.

[21] Yucong Lin and S. Saripalli. Sense and avoid for unmanned aerial vehicles using ads-b. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6402–6407, May 2015.

[22] W. Liu, J. Wei, M. Liang, Y. Cao, and I. Hwang. Multi-sensor fusion and fault detection using hybrid estimation for air traffic surveillance. *IEEE Transactions on Aerospace and Electronic Systems*, 49(4):2323–2339, OCTOBER 2013.

[23] M. Schafer, V. Lenders, and I. Martinovic. Experimental Analysis of Attacks on Next Generation Air Traffic Communication. *Appl. Cryptography Netw. Security*, pages 253–271, 2013.

[24] Mohsen Riahi Manesh and Naima Kaabouch. Analysis of vulnerabilities, attacks, countermeasures and overall risk of the automatic dependent surveillance-broadcast (ads-b) system. *International Journal of Critical Infrastructure Protection*, 19, 2017.

[25] A. Mitrokotsa, C. Dimitrakakis, P. Peris-Lopez, and J. C. Hernandez-Castro. Reid et al.'s distance bounding protocol and mafia fraud attacks over noisy channels. *IEEE Communications Letters*, 14(2):121–123, February 2010.

[26] Jorge Munilla and Alberto Peinado. Distance bounding protocols for rfid enhanced by using void-challenges and analysis in noisy channels. *Wirel. Commun. Mob. Comput.*, 8(9):1227–1232, November 2008.

[27] K. Sampigethaya and R. Poovendran. Security and privacy of future aircraft wireless communications with offboard systems. In *2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011)*, pages 1–6, Jan 2011.

[28] Daniel Baraldi Sesso, Lucio F. Vismari, and JoÃ£o Batista Camargo. An approach to assess the safety of ads-b based unmanned aerial systems. In *Proc. of International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 669–676, 2014.

[29] A. Smith, R. Cassell, T. Breen, R. Hulstrom, and C. Evers. Methods to provide system-wide ads-b back-up, validation and security. In *2006 ieee/aiaa 25TH Digital Avionics Systems Conference*, pages 1–7, Oct 2006.

[30] J. Song, V. W. S. Wong, and V. C. M. Leung. Secure location verification for vehicular ad-hoc networks. In *IEEE GLOBECOM 2008 - 2008 IEEE Global Telecommunications Conference*, pages 1–5, Nov 2008.

[31] Brandon Stark, Brennan Stevenson, and YangQuan Chen. Ads-b for small unmanned aerial systems: Case study and regulatory practices. In *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 152–159, 2013.

[32] M. Strohmeier, V. Lenders, and I. Martinovic. On the security of the automatic dependent surveillance-broadcast protocol. *IEEE Communications Surveys Tutorials*, 17(2):1066–1087, Secondquarter 2015.

[33] uAvionix. uAvionix - Sense and Avoid for Drones and GA. https://uavionix.com/, 2019.

[34] Srdjan Čapkun, Levente Buttyán, and Jean-Pierre Hubaux. Sector: Secure tracking of node encounters in multi-hop wireless networks. In *Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks*, SASN '03, pages 21–32, New York, NY, USA, 2003. ACM.

[35] Kyle D. Wesson, Todd E. Humphreys, and Brian L. Evans. Can cryptography secure next generation air traffic surveillance? http://users.ece.utexas.edu/ bevans/papers/2015/nextgen/, 2014.

[36] Matthias Wilhelm, Jens Schmitt, and Vincent Lenders. Practical message manipulation attacks in ieee 802.15.4 wireless networks. March 2012.

[37] Haomiao Yang, Qixian Zhou, Mingxuan Yao, Rongxing Lu, Hongwei Li, and Xiaosong Zhang. A practical and compatible cryptographic solution to ads-b security. *IEEE Internet of Things Journal*, 2019.

[38] D. H. Yum, J. S. Kim, S. J. Hong, and P. J. Lee. Distance bounding protocol with adjustable false acceptance rate. *IEEE Communications Letters*, 15(4):434–436, April 2011.