On the Feasibility of Exploiting Traffic Collision Avoidance System Vulnerabilities

Paul M. Berges, *Virginia Tech*, pberges7@vt.edu
Basavesh Ammanaghatta Shivakumar, *Purdue University*, bammanag@purdue.edu
Timothy Graziano, *Virginia Tech*, tmgraz@vt.edu
Ryan Gerdes, *Virginia Tech*, rgerdes@vt.edu
Z. Berkay Celik, *Purdue University*, zcelik@purdue.edu

***Abstract*—Traffic Collision Avoidance Systems (TCAS) are safety-critical systems required on most commercial aircrafts in service today. However, TCAS was not designed to account for malicious actors. While in the past it may have been infeasible for an attacker to craft radio signals to mimic TCAS signals, attackers today have access to open-source digital signal processing software, like GNU Radio, and inexpensive software defined radios (SDR) that enable the transmission of spurious TCAS messages. In this paper, methods, both qualitative and quantitative, for analyzing TCAS from an adversarial perspective are presented. To demonstrate the feasibility of inducing near mid-air collisions between current day TCAS-equipped aircraft, an experimental *Phantom Aircraft* generator is developed using GNU Radio and an SDR against a realistic threat model.**

***Index Terms*—TCAS, collision avoidance, aviation security, unauthenticated ranging, safety critical systems**

I. Introduction

Aviation remains the safest way to travel because of the various safety-critical systems operating at any given moment on the aircraft [1]. One such on-board safety feature is known as the Traffic Collision Avoidance System (TCAS), internationally known as the Airborne Collision Avoidance System (ACAS), that prevents the mid-air collisions of transponder-equipped aircraft. In the event of a Near Mid-air Collision (NMAC) where Air Traffic Control (ATC) towers cannot react in time, TCAS is critical for warning pilots to change course and prevent a mid-air collision. Many aviation regulatory bodies mandate the use of TCAS on larger commercial aircraft [2].

Recent technology, such as Software Defined Radio (SDR), enables the manipulation of TCAS through software-defined wireless signals designed to appear like one or more aircraft on a collision course with a target aircraft. TCAS was never intended to perform under adversarial conditions which are entirely feasible for a malicious actor to create in today's environment of inexpensive, powerful computers. If an attacker were to compromise TCAS, they could bypass the safety benefits granted by TCAS equipage. Worse, under certain conditions, a TCAS-equipped aircraft could have a higher chance of mid-air collision than an unequipped aircraft.

Most prior work into the security of the Mode S transponder has been limited to Automated Dependent Surveillance-Broadcast (ADS-B) message spoofing [3], [4] or pilot responses to erroneous TCAS messages but does not specify the technical requirements to spoof TCAS [5]. This paper represents the first research into accurately spoofing TCAS messages, and the danger of induced near mid-air collisions as a result of the aforementioned spoofed messages. TCAS and ADS-B are closely linked because both messages are transmitted through the Mode S (Selective Aeronautical telecommunication; able to be interrogated) transponder. While works on ADS-B, propose some defenses against spoofing attacks, ADS-B is not part of any safety-critical systems on an aircraft, negating the need for sweeping and expensive security changes. If TCAS is also shown to be vulnerable to attack, then the safety of onboard passengers would be compromised, providing a more significant motivator to institute system changes and improvements. Due to the safety implications of this research, our work was originally done in conjunction with a primary manufacturer of TCAS and with notification to Department of Homeland Security (DHS).

In this paper, we demonstrate that the TCAS transactions are vulnerable to attack. We explore TCAS vulnerabilities using the weakest possible attack model using only open-source software, publicly-accessible knowledge about TCAS, and a low cost SDR. We hope that our explorations motivate further research into the defense of TCAS and Mode S transponders. In this work, we make the following contributions:

- We show that a safety-focused design approach for safety-critical systems does not result in a safe system in an adversarial environment.
- We take an analytical approach to find vulnerabilities in, and explore the effect of failures of, TCAS II through an attack tree.
- We develop a Phantom Aircraft attack, where we use open-source software to spoof critical TCAS II components (e.g., a Mode S Transponder) enabling an attacker to masquerade as a collision-bound aircraft.

II. Background

We provide background on TCAS and methods for attack construction to aid understanding and evaluation of the Phantom Aircraft attack provided in the subsequent sections.

A. Traffic Collision Avoidance System (TCAS)

TCAS is designed to reduce the risk of Near Mid-air Collisions (NMAC) between aircraft. It operates independently of ATC to notify/instruct pilots when a protected volume of airspace around the aircraft is intruded upon by other aircraft. It uses an onboard transponder (transmitter/receiver) to facilitate air-to-air communications between aircraft. Each TCAS interrogates nearby aircraft on the 1030 MHz frequency band and all aircraft reply on the 1090 MHz frequency band. TCAS uses these transactions to track nearby aircraft and monitor for intrusions in nearby airspace. Figure 1 shows the protected region generated by TCAS defined by altitude and the remaining time until an intruder reaches its Closest Point of Approach (CPA) [6], [7]. TCAS is designed to operate independently of ATC and other tracking systems in order to mitigate collision risk regardless of the operational state of other systems or ground based information relayed to the pilot.

TCAS System Components. TCAS requires that aircraft be equipped with a transmitter/receiver known as a Mode S

978-1-7281-4760-4/20/$31.00 ©2020 IEEE

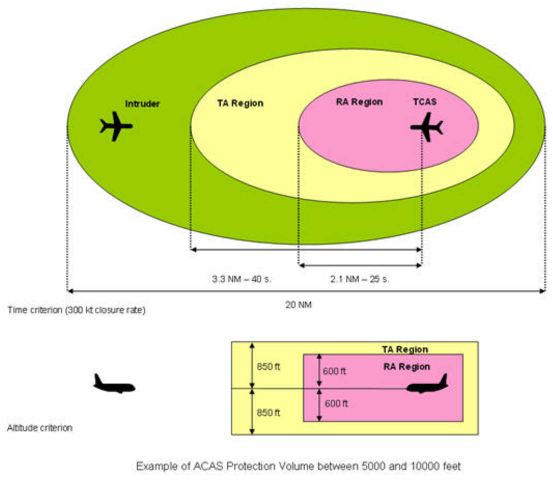


Fig. 1. The region protected by TCAS [6].

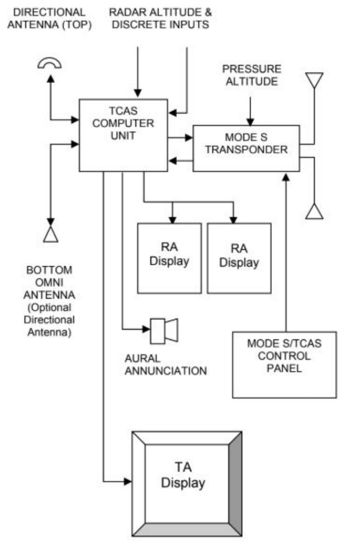

DIRECTIONAL ANTENNA (TOP)
RADAR ALTITUDE & DISCRETE INPUTS
PRESSURE ALTITUDE
TCAS COMPUTER UNIT
MODE S TRANSPONDER
BOTTOM OMNI ANTENNA (Optional Directional Antenna)
RA Display
RA Display
AURAL ANNUNCIATION
MODE S/TCAS CONTROL PANEL
TA Display

Fig. 2. TCAS System Components [7]

transponder. TCAS itself consists of three main components illustrated in Figure 2:

- **TCAS CPU:** A computer that performs interrogations, surveillance, tracking, threat declaration, etc.
- **Antennas:** There are three required antennas and one optional antenna. However, many implementations choose a shared antenna design to save costs.
- **Cockpit displays:** Information on intruding aircraft and suggested responses are visually and aurally announced to pilots.

TCAS: Request-Response. Generally, TCAS behaves within a request-response architecture. A TCAS CPU uses the 1030 MHz channel to request information from an intruder. This action is called an *interrogation*. The Mode S transponder on an intruding aircraft is responsible for responding to interrogations [8], [9]. Each plane has a unique 24-bit address called its International Civil Aviation Organization (ICAO) address [8], [10]. All interrogations are addressed to the aircraft receiving the interrogation. ICAO addresses are combined using an exclusive-or function with a unique Mode S cyclic redundancy check (CRC) calculation to form the Address/Parity (AP) field [11]. The data encoded in interrogation is referred to as an Uplink Format (UF) message. UF messages have a fixed-length 5-bit header, and its value indicates what type of UF message it is. Replies are broadcast on the 1090 MHz frequency band and only happen if an interrogation's decoded AP field matches the ICAO address of the receiving aircraft. The data encoded in a reply is referred to as a Downlink Format (DF) message. The DF header's value will always match the UF header's value of the message that started the transaction.

TCAS advisories. There are two types of advisories (i.e., responses to intrusions) that TCAS can produce.

- A **Traffic Advisory (TA)** is intended to help a pilot visually locate an intruding aircraft; it is issued first.
- A **Resolution Advisory (RA)** will recommend maneuvers or positional holds in order to increase the separation between the aircraft and intruder [7].

In a coordinated maneuver, the aircraft that announces its RA first controls the maneuvers of the second aircraft. In other words, the aircraft that makes the first decision will coordinate the possible maneuvers the other aircraft can do. In the event of simultaneously received RA coordination, the lower ICAO identification number receives precedence. An attacker can control this within the spoofed message to always assert primacy. Then, the TCAS will choose the maneuver's *sense* and *strength* [7]. A sense is the direction the aircraft will maneuver. After a sense selection, TCAS will choose a maneuver strength. A strength can either be positive or negative. Upon receipt of a RA, the pilot is obliged to follow the RA maneuver guidance if possible, giving higher priority to the RA than ATC guidance. Related work has shown that pilots will follow an RA even if they believe the system to be behaving inappropriately or "crying wolf" [5].

TCAS Operating Modes. TCAS can be operated in three modes, which are controlled by the pilot [7]. The level of functionality TCAS provides is defined by its Sensitivity Level (SL). These levels are:

- (SL 1) **Standby**: No TCAS tracking or "squitters" (a non-solicited Mode S transmission of aircraft tracking data to alert nearby transponders of the aircraft's presence). Mode S transponder will respond to interrogations.
- (SL 2) **TA Only**: TCAS will track intruding aircraft, but it will only announce a TA to the pilot. Mode S transponder transmits squitters and responds to interrogations.
- (SL 3+) **TA-RA**: TCAS will automatically select the best SL for the altitude.

In some instances ATC towers can partially control TCAS. Specifically, ground controllers can place TCAS into SL 2 or greater [7], [8], [10]. Only after repeated TCAS malfunctions will a pilot degrade the SL [5].

B. Software-defined Radio

A Software-defined radio (SDR) does not use dedicated hardware circuits for signal processing but instead conducts radio-relevant processing in software providing flexibility in terms of modulation, filtering, operating frequency, and frame format [12]. GNU Radio is a free and open-source development

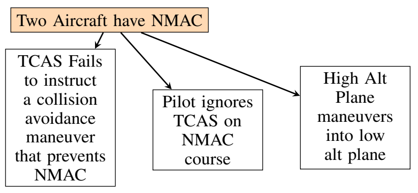

Two Aircraft have NMAC
TCAS Fails to instruct a collision avoidance maneuver that prevents NMAC
Pilot ignores TCAS on NMAC course
High Alt Plane maneuvers into low alt plane

Fig. 3. The root and major leaf nodes of the TCAS attack tree

platform for SDR that uses a graphical approach to radio design and supports development in C++ or Python [13]. It is often used in a simulation environment to support real-world radio systems and wireless communications research [14]. GNU Radio Companion (GRC) is a graphical application packaged with GNU Radio [13].

C. Attack Tree and Fault Tree Analysis

An attack tree is a graphical representation of an attack including simply a series of "What if ..." questions that result is an extensive list of known attack vectors to the system [15]. Fault Tree Analysis (FTA) is a tool that safety engineers use to inform decision making concerning design revision with respect to some undesired event [16]. FTAs are created to model failure in very complex systems and can show how a series of events can lead to a larger undesired failure event using probability, Boolean algebra, etc. Similar to the attack tree, the root of an FTA starts from a goal (i.e., starting from the desired failure to investigate) and then deduces the specific components that could cause the failure.

III. TCAS Attack Tree

We consider an attacker who aims to induce an NMAC. The attack tree of this work shares two major components with the TCAS fault tree; the major components identified are shown as the top of the attack tree in Figure 3. The left event represents *unresolved NMAC* attacks (e.g., two planes are already on a collision course, and the pilots fail to maneuver the planes in a way that avoids NMAC), and the right event represents the *induced NMAC* attacks (e.g., two planes that were not previously on course towards NMAC are maneuvered into an NMAC). The central event accounts for any attacks on people like social engineering [17]. Our particular interest is the induced component of NMAC because it is antithetical to the design goals of TCAS and implies that an attacker could intentionally redirect two nearby planes into each other.

Induced NMAC Attacks. Induced NMAC attacks are reliant on fooling TCAS into generating an RA that, if followed, would induce an NMAC. A section of the induced NMAC attack is shown in Figure 4. Attacks that would induce an NMAC require that the expected Mode S messages are obeyed to trick a TCAS into a *false track*, i.e., following a non-existent *Phantom Aircraft* course, whose purpose is to cause the target aircraft to maneuver onto a probable collision course with another aircraft. TCAS implicitly trusts that responses to its interrogations originate from an actual aircraft and that the aircraft will travel in a certain manner. If an attacker can respond to TCAS's interrogations and appear to move like a plane, then the false track should be successfully created and maintained. The foundation of the Phantom Aircraft starts with an accurate spoofer of Mode S signals; it must also consistently

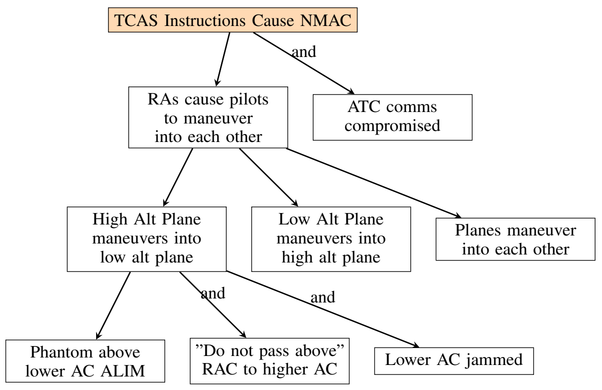

TCAS Instructions Cause NMAC
and
RAs cause pilots to maneuver into each other
ATC comms compromised
High Alt Plane maneuvers into low alt plane
Low Alt Plane maneuvers into high alt plane
Planes maneuver into each other
and
and
Phantom above lower AC ALIM
"Do not pass above" RAC to higher AC
Lower AC jammed

Fig. 4. A simplified version of the branch on induced NMAC attacks.

fool the range measurements that TCAS performs [8], [18]. The induced NMAC attacks are reliant on two threats being near in altitude to increase the probability of a successful attack.

IV. Phantom Aircraft Attack

We develop a particular induced NMAC attack: the Phantom Aircraft attack. First, we define the threat model and outline the attack model. This represents a specific example of a TCAS attack. The design of any attack in the attack tree of Section III would follow similar steps to the procedure shown.

Threat Model. The attacker's goal is to induce an NMAC between two aircraft without any modification to the TCAS on the aircraft. The attack therefore requires replying to relevant interrogations as well as emulating *distance closing* signal behavior, in which the Round Trip Time (RTT) range estimation of TCAS is fooled with proper response timing into believing that an aircraft is intruding. The attacker launches the attack using GNU Radio [13] and a relatively low-cost SDR hardware from Ettus Research [19]. In a short list, the attacker's capabilities are the following:

- The attacker is equipped with an SDR and directional antennas capable of transmitting signals to the targeted aircraft.
- The location and speed of the targeted aircraft are known to the attacker.
- The attacker is able to selectively jam Mode S transmissions of aircraft using knowledge of their ICAO address and its reply periodicity. Directional jamming is not required given this knowledge.
- The attacker can properly time their replies, with respect to the interrogation periodicity of an aircraft, to appear closer or farther than they really are.

Attack Model. The intended encounter is the scenario shown in Figure 5. For analysis two aircraft are approaching each other without any horizontal offset. These aircraft are also approaching head-on such that there is an exact 180-degree difference between their bearings with no vertical rate of approach. Alternate approach angles do not impact the feasibility of the attack. The aircraft are placed such that they have the adequate vertical separation that would not elicit any TCAS warnings. The attacker begins selectively jamming the black aircraft (the lower one on the left in Figure 5) to drop its track from the target aircraft. A false track is baited, and maintained,

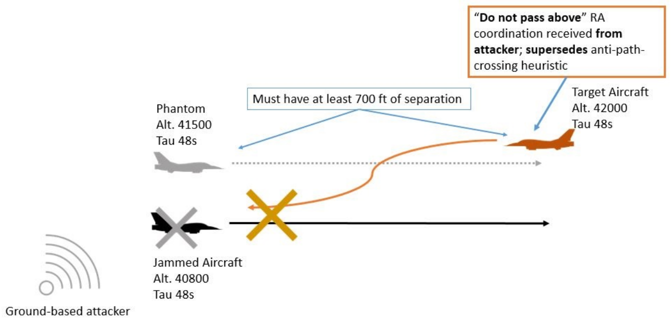

"Do not pass above" RA coordination received from attacker; supersedes anti-path-crossing heuristic
Must have at least 700 ft of separation
Phantom
Alt. 41500
Tau 48s
Target Aircraft
Alt. 42000
Tau 48s
Jammed Aircraft
Alt. 40800
Tau 48s
Ground-based attacker

Fig. 5. Planned Phantom Aircraft attack against two nearby intruders

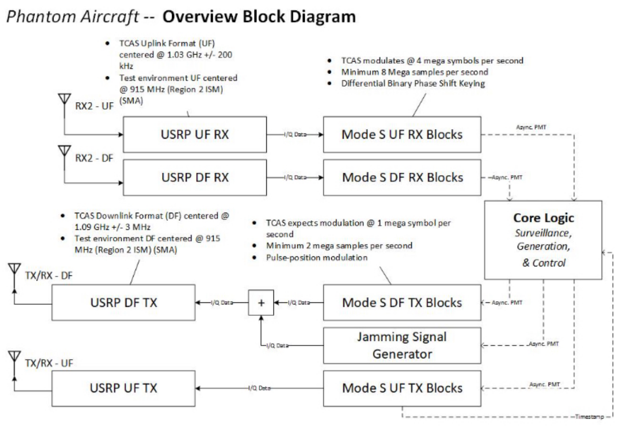

Phantom Aircraft -- Overview Block Diagram
TCAS Uplink Format (UF) centered @ 1.03 GHz +/- 200 kHz
Test environment UF centered @ 915 MHz (Region 2 ISM) (SMA)
TCAS modulates @ 4 mega symbols per second
Minimum 8 Mega samples per second
Differential Binary Phase Shift Keying
RX2 - UF
USRP UF RX
I/Q Data
Mode S UF RX Blocks
Async. PMT
RX2 - DF
USRP DF RX
I/Q Data
Mode S DF RX Blocks
Async. PMT
Core Logic
Surveillance, Generation, & Control
TCAS Downlink Format (DF) centered @ 1.09 GHz +/- 3 MHz
Test environment DF centered @ 915 MHz (Region 2 ISM) (SMA)
TCAS expects modulation @ 1 mega symbol per second
Minimum 2 mega samples per second
Pulse-position modulation
TX/RX - DF
USRP DF TX
I/Q Data
+
I/Q Data
Mode S DF TX Blocks
Async. PMT
I/Q Data
Jamming Signal Generator
Async. PMT
TX/RX - UF
USRP UF TX
I/Q Data
Mode S UF TX Blocks
Async. PMT
Timestamp

Fig. 6. Overview block diagram of the phantom aircraft generator.

by the attacker on the target aircraft. By analyzing the expected RTT of the request-response TCAS handshake, an attacker can manipulate artificial latency by delaying their reply once the interrogation is received, in order to appear to be at a desired distance and closure rate. This calculation requires knowledge of the round trip distance between the victim and phantom times the speed of light, factoring the computational latency of a TCAS module upon message receipt, and the ability for the attacker to transmit spoofed messages with ns timing resolution.

Once the attacker has determined that the target is within the TA range of the phantom, the phantom must send its Resolution Advisory Complement (RAC) first. The attacker can simply announce an RA arbitrarily because TCAS trusts that the RAC it receives is from a transponder acting in good faith. The attacker forces the target TCAS to cross paths with the phantom by constructing a "Do not pass above" coordination [10]. In combination with the required Altitude Limit (ALIM) for this flight level, the target would need to descend to an altitude no greater than 40,800 ft.

To accomplish the Phantom Aircraft attack, five conditions must be met. The attacker needs to:

1) Perform reconnaissance by interrogating the airspace in the attacker's vicinity.
2) Estimate the trajectory of victim aircraft through tracking.
3) Bait a false track from the victim by emitting squitters and detecting them when interrogations begin.
4) Maintain the false track by responding to interrogations.
5) Declare a threat against the victim and detect evidence that the victim declared its own RA.

V. Attack Implementation

We developed an attack platform based on GNU Radio. We present the block diagrams of the platform and simulation methods. Lastly, an outline of the hardware used for this platform is presented.

A. Phantom Aircraft Generator

We demonstrate a proof-of-concept phantom aircraft generator for GNU Radio application to bait a TCAS into tracking a phantom aircraft by having a pair of zero-speed aircraft track each other. The core functionality of an attacker and a normal aircraft are identical. Figure 6 shows the overview of the phantom aircraft generator as well as expected carrier frequencies, modulations, and sample rates.

We designed the system using open-source GNU Radio modules. The reception and demodulation of DF packets are handled by *gr-adsb* (ADS-B OOT for GNU Radio) [20]. The total OOT dependencies for gr-modes are: (1) gr-adsb: demodulation and decoding of ADS-B messages [20], (2) gr-burst: blocks for building PSK modems [21], (3) gr-eventstream [22], (4) gr-mapper [23], and (5) gr-pyqt [24].

Universal Software Radio Peripheral (USRP) hardware requires significant processing power in general, especially at high sample rates. The PC for the attack uses an Intel Core i7-6800K; six cores at a 3.4 GHz clock rate with 16GB RAM. An Ettus Research B210 is the transmitting USRP, and an Ettus Research N210 is the receiving USRP [25], [26]. These USRPs are low cost, $1,259 and $2,011 respectively. The two USRPs communicate over SMA cables through a variable attenuator that prevents the receiver's analog to digital converter (ADC) from being saturated.

B. Phantom Aircraft Attack Details

We present the architecture of the phantom aircraft generator, the sub-routines of each high-level block, and the interfaces between each function.

Our system uses both asynchronous and streamed data. A TX and RX interface for both interrogations and replies is used. A nearby carrier frequency in the industrial, scientific, and medical (ISM) band is chosen for testing [27] instead of normal TCAS frequency bands [7], [8], [10]. Each RX interface converts raw samples into packets the Core Logic can understand; each TX interface performs the opposite. The range estimation occurs from the RTT of the interrogation-reply cycle using timestamp feedback from the TX and RX blocks.

Conversion of data from the Core Logic to samples interface also handles the packing of message data into a complete Mode S message. A lambda function block from gr-pyqt and a prepend preamble block from gr-burst are convenient for packing the data into a Mode S reply [21], [24].

The data that enters the DF TX interface from the Core Logic is 56 or 112 samples of bytes in length. To convert this message into a Mode S reply, the following procedure is performed: (a) Pulse Position Modulation (PPM) is performed [28], (b) the Mode S reply preamble is attached to the beginning of the packet, and (c) the packet is interpolated to the application's base sample rate. Then, the packet must be converted to a *stream* of complex samples the GNU Radio supports. Once the payload is packed into a complete Mode S reply, gr-eventstream is used to insert the data from an *asynchronous* packet into the sample stream.

The receiver interface for DF messages is made from gr-adsb blocks [20]. The ADS-B Framer is slightly modified such that it creates a timestamp that propagates to the Core Logic when a preamble is detected. The ADS-B Framer block correlates the input with the expected preamble and creates a tag object on the first sample of the preamble when a match is detected. This tag serves as the synchronization point which the ADS-B Demodulator can use for PPM demodulation. We mimic the design of the reply transmission interface such that the interface remains independent of actual message data. Some adjustments are made to support the different pulse sequence and modulation scheme of Mode S interrogations and to suppress Modes A and C transponders from processing the interrogation.

The receiving interface for the Interrogation Framer block performs correlation analysis of the preamble and creates a tag on the first sample of the preamble. Using this as a synchronization point, the Tag Consumer block then creates a "slice" of samples from 1-120 Differential Binary Phased Shift Keyed (DBPSK) samples (i.e., length of the preamble + 112 symbols assumed). Demodulation is performed asynchronously. The Core Logic interfaces then convert interrogation and reply samples into a format the state machine logic can understand.

VI. Evaluation

We verify the attack detailed in Section V. While the direct testing approach would be to generate aircraft on a real TCAS, real TCAS hardware is not trivial to acquire and setup. Therefore, verification is completed with two simplified aircraft cores that can track each other at zero speed through the medium of Mode S messages. We performed a series of unit and integration tests through GNU Radio to verify the adversarial tasks. For each point of verification below, we executed the flowgraph for 10 minutes with a fixed surveillance period of one second such that approximately 600 rounds of interrogations and replies occur for each item. We have verified the following:

1) Arbitrary Mode S reply messages can be crafted to fool the gr-adsb receiver without USRP hardware in-the-loop [20]).
2) The gr-adsb receiver's performance is characterized via a packet-loss calculation with USRP hardware-in-the-loop.
3) Arbitrary Mode S interrogation messages can be crafted to fool a custom Mode S interrogation receiver without USRP hardware-in-the-loop.
4) All four message modulators and demodulators can deliver payloads to the core logic. The core logic performance is indicative of the state machine functionality. No USRP hardware is used in-the-loop.

Mode S Reply. We first verified that we correctly modulated, demodulated, and decoded a crafted payload message as shown in Figure 7. Six different 10-minute test runs are performed. We found that even the best performing case still has a packet-loss factor nearly 60%. While this is a significant problem for hardware testing, our primary focus is its emulation of TCAS functionality. Therefore, the remainder of the tests ignores the use of USRP hardware.

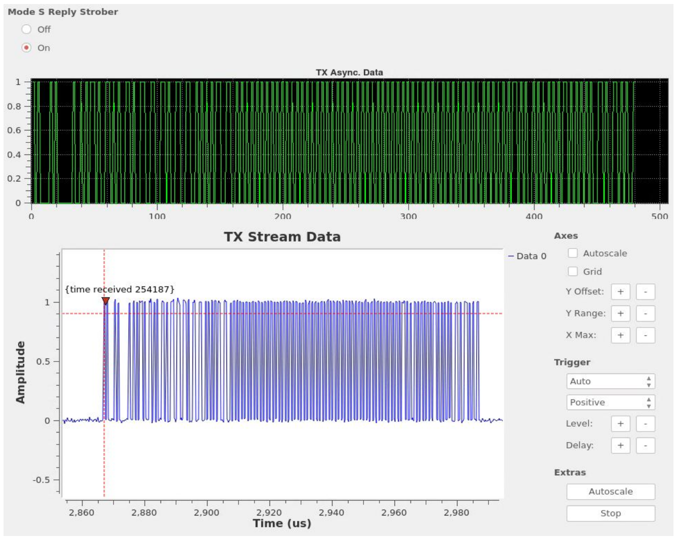


Fig. 7. Real-time graph showing the DF 17 message received and decoded.

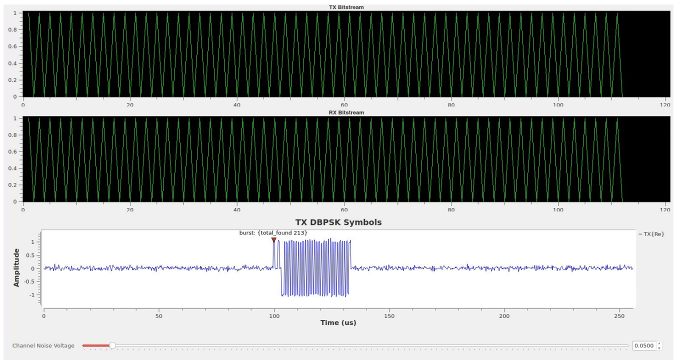


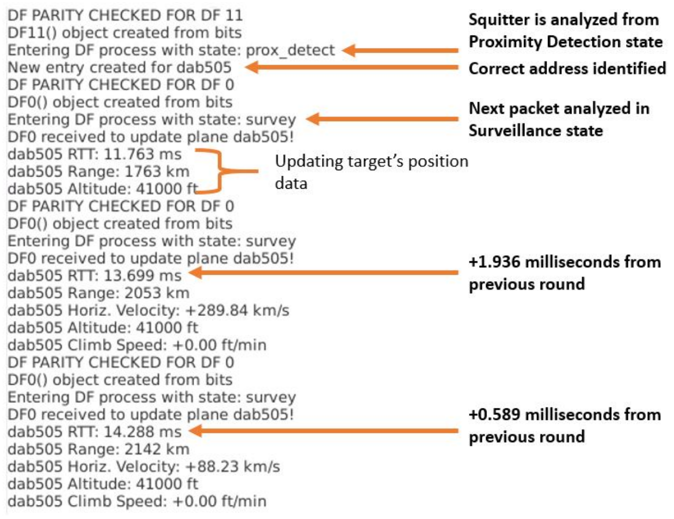
Fig. 8. Three real-time plots of the interrogation interface test. The top plot shows the asynchronous TX bit stream, the middle plot shows the asynchronous RX bit stream, and the lower plot shows the real-time plot of the framer marking the start of the message.

Mode S interrogation. We verified that an arbitrary payload of 112 bits is modulated and demodulated to confirm an expected Mode S interrogation through a simulated Additive White Gaussian Noise (AWGN) channel as shown in Figure 8.

Core Logic. We verified that the core logic demonstrates required states in software-only environment, proving the base-level functionality of this phantom aircraft generator. We note that altitude information is as expected. Packet throughput and latency are evaluated as they pertain to maintaining the phantom aircraft's tract in the victim's TCAS with the correct range estimation. However, the computational overhead is too large and imprecise for range estimation, as shown in Figure 9. Further investigation into improving the RTT estimation and packet throughput is required.

VII. Discussion and Limitations

Based on our experience, the implementation of a phantom aircraft generator will require the removal of expensive message interfaces between modules and the implementation of high-accuracy time-stamping in the code.



```
DF PARITY CHECKED FOR DF 11
DF11() object created from bits
Entering DF process with state: prox_detect
New entry created for dab505
DF PARITY CHECKED FOR DF 0
DF0() object created from bits
Entering DF process with state: survey
DF0 received to update plane dab505!
dab505 RTT: 11.763 ms
dab505 Range: 1763 km
dab505 Altitude: 41000 ft
DF PARITY CHECKED FOR DF 0
DF0() object created from bits
Entering DF process with state: survey
DF0 received to update plane dab505!
dab505 RTT: 13.699 ms
dab505 Range: 2053 km
dab505 Horiz. Velocity: +289.84 km/s
dab505 Altitude: 41000 ft
dab505 Climb Speed: +0.00 ft/min
DF PARITY CHECKED FOR DF 0
DF0() object created from bits
Entering DF process with state: survey
DF0 received to update plane dab505!
dab505 RTT: 14.288 ms
dab505 Range: 2142 km
dab505 Horiz. Velocity: +88.23 km/s
dab505 Altitude: 41000 ft
dab505 Climb Speed: +0.00 ft/min
```

Squitter is analyzed from Proximity Detection state
Correct address identified
Next packet analyzed in Surveillance state
Updating target's position data
+1.936 milliseconds from previous round
+0.589 milliseconds from previous round

Fig. 9. Imprecise RTT measurements from the design under test

Heretofore the nature of TCAS, as a highly complex system, has allowed it to enjoy a sort of security through obscurity" [29]. We have, however, shown that a relatively low-resourced attacker can reproduce the essential signals of TCAS so as to mount an attack against it. Those with the time and resources to attack a complicated system will eventually succeed. It is imperative for defenders to act now in securing the integrity of these systems.

As future work, it is also essential to develop a platform in which implementations can be tested on real TCAS hardware. TCAS cannot operate independently from its inputs; it has subroutines which disable the TCAS if inputs are failing or nonexistent [7], [8], [10]. Therefore, designing a test bench in which TCAS can operate on real or simulated inputs would be invaluable in conducting accurate security research. Future attack models will focus on a more specific approach and conduct the spoofed message calculation and modulation for expected attack packets beforehand to reduce latency. The degree in which an attacker could control airspace given a fully functional phantom aircraft generator will be also explored.

VIII. Conclusions

We presented an exploration of TCAS vulnerabilities to exploitation. A wide breadth of analysis is done with respect to TCAS security with the intention to motivate academic and industry-led research into the security of safety-critical airborne collision avoidance systems. To accomplish this goal, the following actions are taken. First, a TCAS safety study is analyzed from an adversarial perspective to quantify the effect of attacks on the overall NMAC risk ratio. Second, attacks on TCAS are explored through the model of an attack tree. Third, an attack is chosen, and a threat model is defined to relay the attacker's goals and capabilities. Finally, an implementation of a threat using GNU Radio is presented, and critical components of the implementation are tested.

References

[1] C. Ingraham, "The safest and deadliest ways to travel," https://tinyurl.com/y58zm5p2, May 2015, accessed 2019-03-26.

[2] EUROCONTROL, "ACAS II equipage requirements," https://tinyurl.com/y2hjz64j, March 2010, accessed 2019-03-26.

[3] H. Yang, M. Yao, Z. Xu, and B. Liu, "Lhcsas: A lightweight and highly-compatible solution for ads-b security," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, Dec 2017, pp. 1–7.

[4] P. Berthier, J. M. Fernandez, and J. Robert, "Sat : Security in the air using tesla," in *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, Sep. 2017, pp. 1–10.

[5] J. H. V. L. Matthew Smith, Martin Strohmeier and I. Martinovic, "A view from the cockpit: Exploring pilot reactions to attacks on avionic systems," http://www.cs.ox.ac.uk/files/11581/ndss-2020.pdf, March 2020, accessed 2020-03-29.

[6] SKYbrary, "Airborne collision avoidance system (acas)," EUROCONTROL, Tech. Rep., 2019, accessed 2019-04-01.

[7] *Introduction to TCAS II Version 7.1*, Federal Aviation Administration (FAA), February 2011, accessed 2019-03-28.

[8] *Airborne Collision Avoidance System (ACAS) Manual*, 1st ed., International Civil Aviation Organization (ICAO), 2006, accessed 2019-04-02.

[9] D. Officer, "Secondary surveillance radar mode s advisory circular," International Civil Aviation Organization (ICAO), 1000 Sherbrooke Street West, Suite 400, Montreal, Quebec, Canada, Tech. Rep., 1983, cIRCULAR 174-AN/110.

[10] *Aeronautical Telecommunications: Annex 10 to the Convention on International Civil Aviation, Volume IV Surveillance and Collision Avoidance Systems*, 4th ed., International Civil Aviation Organization (ICAO), July 2007, accessed 2019-04-02.

[11] P. Hunt, *CRC Calculation For MODE-S Transponders*, EuroControl, November 1994, accessed 2019-04-02.

[12] J. Mitola, "Software radios," *IEEE Communications magazine*, vol. 33, no. 5, pp. 24–25, 1995.

[13] T. G. R. Foundation, "Gnu radio - the free & open source radio ecosystem," https://www.gnuradio.org/, 2019, accessed 2019-04-09.

[14] E. Blossom, "Exploring GNU radio," *http://www. gnu. org/software/gnuradio/doc/explorin g-gnuradio. html*, 2004, accessed 2019-04-01.

[15] B. Schneier, "Attack trees," https://tinyurl.com/y8x686mt, 1999, accessed 2019-04-04.

[16] C. A. E. II, "Fault tree analysis," University of Central Florida, Tech. Rep., 1999, accessed 2019-04-04.

[17] *What is social engineering? Tips to help avoid becoming a victim*, https://tinyurl.com/y5fawajm, Symantec Corporation, 2018, accessed 2019-04-07.

[18] C. H. Kim, G. Avoine, F. Koeune, F.-X. Standaert, and O. Pereira, "The swiss-knife rfid distance bounding protocol," in *Information Security and Cryptology – ICISC 2008*, 2008, pp. 98–115.

[19] N. Instruments, "Ettus research - the leader in software defined radio (sdr)," https://www.ettus.com/, 2019, accessed 2019-04-09.

[20] M. Hostetter, "gr-adsb: Gnu radio oot module for demodulating and decoding ads-b packets," https://github.com/mhostetter/gr-adsb, July 2018, accessed 2019-04-11.

[21] T. O'Shea and K. Karra, "gr-burst: Burst psk modem building blocks for gnu radio," https://github.com/gr-vt/gr-burst, October 2016, accessed 2019-04-11.

[22] T. O'Shea, "gr-eventstream," https://github.com/osh/gr-eventstream, April 2017, accessed 2019-04-11.

[23] T. O'Shea and K. Karra, "gr-mapper: Symbol to bit mapping and demapping blocks for gnu radio," https://github.com/gr-vt/gr-mapper, October 2018, accessed 2019-04-11.

[24] T. O'Shea, "gr-pyqt: pyqt based plotters intended for plotting bursted events in gnu radio," https://github.com/osh/gr-pyqt, July 2016, accessed 2019-04-11.

[25] *USRP Hardware Driver and USRP Manual: USRP B2x0 Series*, http://files.ettus.com/manual/page_usrp_b200.html, National Instruments, 2018, accessed 2019-04-13.

[26] *USRP Hardware Driver and USRP Manual: USRP2 and N2x0 Series*, http://files.ettus.com/manual/page_usrp2.html, National Instruments, 2018, accessed 2019-04-13.

[27] *Terms and definitions*, http://life.itu.int/radioclub/rr/art1.pdf, International Telecommunication Union, 2009, accessed 2019-04-16.

[28] Y. Fujiwara, "Self-synchronizing pulse position modulation with error tolerance," *IEEE Transactions on Information Theory*, vol. 59, no. 9, pp. 5352–5362, Sep. 2013.

[29] D. Hayes, "Security through obscurity is not security at all," https://wpshout.com/security-through-obscurity/, October 2017, accessed 2019-04-23.