SECURITY IMPROVEMENTS FOR THE AUTOMATIC IDENTIFICATION SYSTEM

by

Robert E. Litts
B.S. May 2013, United States Coast Guard Academy

A Thesis Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

ELECTRICAL AND COMPUTER ENGINEERING

OLD DOMINION UNIVERSITY
May 2021

Approved by:

Dimitrie C. Popescu (Director)

Otilia Popescu (Member)

Linda Vahala (Member)

ABSTRACT

SECURITY IMPROVEMENTS FOR THE AUTOMATIC IDENTIFICATION SYSTEM

Robert E. Litts
Old Dominion University, 2021
Director: Dr. Dimitrie C. Popescu

The Automatic Identification System (AIS) is used aboard the vast majority of sea-going vessels in the world as a collision avoidance tool. Currently, the AIS operates without any security features, which make it vulnerable to exploits such as spoofing, hijacking, and replay attacks by malicious parties. This paper examines the work that has been done so far to improve AIS security, as well as the approaches taken on similar problems in the aircraft and vehicular mobile ad-hoc network (MANET) industries. The first major contribution of this paper is the implementation of a Software Defined Radio (SDR) AIS transmitter and receiver which can be used to conduct vulnerability analysis and test the implementation of new security features. The second contribution is the design of a novel authentication protocol which overcomes the existing vulnerabilities in the AIS system. The proposed protocol uses time-delayed hash-chain key disclosures as part of a message authentication code (MAC) appended to automatic position reports to verify the authenticity of a user. This method requires only one additional time slot for broadcast authentication compared to the existing standard and is a significant reduction in message overhead requirements compared to alternative approaches that solely rely on public key infrastructure (PKI). Additionally, there is an embedded time stamp, a feature lacking in the existing system, which makes this protocol resistant to replay attacks. A test implementation of the proposed protocol indicates that it can be deployed as a link layer software update to existing AIS transceivers and can be deployed within the current AIS technical standards as an expanded message set.

Dedicated to my wife Kristy, for humoring me by listening to me to talk about my research during our daily walks. I hope our son enjoyed hearing my rants and will one day read this and recall those memories as well. And to my mother, you really missed out getting to listen to me talk about this paper! I know you have been helping me every step of the way just as you have always done.

ACKNOWLEDGEMENTS

I would like to thank Dr. Gary Kessler, visiting professor at the U.S. Coast Guard Academy, for taking the time out of his retirement to discuss AIS with me, review my work, and put me in touch with several others who are working on AIS projects throughout the Coast Guard. Your website, papers, and book provided me with the inspiration I needed to get started and keep working on this project. Additionally, I would like to thank Lee Luft, Chairman of the NMEA AIS Working Group for speaking with me about AIS security and future plans for the system. Finally, I would like to thank my advisor, Dr. Dimitrie Popescu for helping me from the very beginning and throughout my journey at ODU. I enjoyed learning from you over the past two years on course work, projects, and research. You gave me the freedom to work on what I enjoyed while challenging me to improve my knowledge.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

In 1989, the tanker vessel Exxon Valdez spilled 11 million gallons of oil into Alaska's Prince William Sound, a disaster which cost over $7 billion to clean and severely disrupted the local population for many decades. One result of this disaster was the 1990 Oil Pollution Act (OPA) which required the U.S. Coast Guard to improve vessel tracking and monitoring services within ports and harbors similar to aviation air traffic controllers [3]. Over the next decade, the international community worked on various systems to meet such a need, and ultimately the International Telecommunication Union (ITU) and International Maritime Organization (IMO) decided that a standardized protocol for international usage would be a benefit to the maritime community [4]. In the late 1990s, the Automatic Identification System (AIS) was created as a situational awareness and collision avoidance tool to provide Vessel Traffic Services (VTS) with improved clarity in harbors and improve navigational safety onbord vessels operating in these often-chaotic sea lanes. However, this system was created in a pre-9/11 world when cybersecurity was not a requirement, so the system operates freely within the maritime Very High Frequency (VHF) band [5]. AIS was created with the assumption that all users would operate with respect and would not attempt to use this tool for nefarious purposes. To that degree, the maritime community has been lucky. In the early 2000s, AIS became mandatory onboard the vast majority of commercial vessels and gave them a complete electronic picture of all surrounding vessels regardless of the weather, sea state or visibility, which commonly cause RADAR deterioration. Today, the availability of small, inexpensive AIS transceivers means that almost every vessel on the ocean operates with AIS.

The International Maritime Organization's (IMO) Safety of Life at Sea (SOLAS) agreement formally dictates the type and size of vessels that are required to carry an operational AIS system. The IMO adopted the SOLAS agreement following the sinking of the Titanic in 1914 and has subsequently updated this international treaty as technology has expanded in order to ensure widespread safety practices are being carried out on the high seas. SOLAS chapter V contains safety of navigation information and specifically lists requirements for shipborne navigational systems and equipment. SOLAS Chapter V Regulation 19.2.4 states that "All ships of 300 gross tonnage and upwards engaged on international voyages

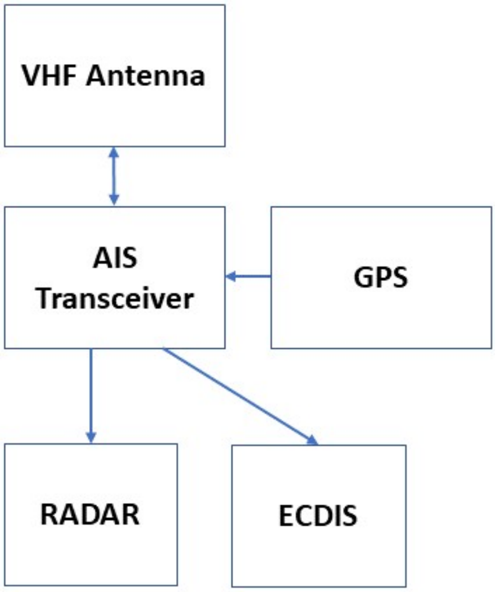


Fig. 1: Typical AIS Configuration aboard a vessel connected to electronic navigation equipment

and cargo ships of 500 gross tonnage and upwards not engaged on international voyages and passenger ships irrespective of size shall be fitted with an automatic identification system (AIS)" [6]. In the United States, the Code of Federal Regulations (CFR) Title 33 §164.46 expands SOLAS V and requires additional vessels to carry AIS Class A/B devices within U.S. waters. CFR 33 also requires vessels to accurately broadcast a properly assigned maritime mobile service identity (MMSI) number and upkeep all AIS data fields and system updates [7]. SOLAS Chapter V 18.9 requires an annual test by an approved surveyor or testing facility that verifies the correct programming of static information and on-air RF testing. Guidelines for this annual test are covered in the IMO's Maritime Safety Committee (MSC) Circular 1252 [8]. Although there are no U.S. regulations that require a qualified inspection of AIS in U.S. navigable waters, the U.S. Coast Guard publishes an AIS Inspection Checklist and Report that mirrors the IMO testing as well as a detailed AIS encoding guide that walks vessel operators through the proper input of each AIS parameter [9, 10]. A typical AIS setup onboard a vessel would include connections to other navigation sensors is shown in Fig. 1.

The VHF antenna allows the AIS transceiver to send/receive AIS messages to other vessels. Dynamic information about a vessel is automatically input from the ship's GPS which provides position, course, and speed data. This information is then fed into the ships electronic chart display information system (ECDIS) which can overlay AIS and radar information on a navigation chart to improve situational awareness. It is important to note that although dynamic positional data is directly fed from the GPS unit, static AIS information must be manually set up and maintained by the vessel operators.

1.1 AIS SECURITY

Neither the ITU M.1371-5 nor the IMO SOLAS standards implicitly include message confidentiality, integrity, or authentication of participating users; therefore, AIS lacks some of the fundamental principles of a secure network. Just as the early days of the internet assumed all users would act with good intentions, AIS was initially created with those same hopes. The U.S. Coast Guard Navigation Center (NAVCEN) frequently asked questions page states that "AIS by design, is an open, non-proprietary, unencrypted, unprotected radio system, intended to operate on non-secure VHF-FM channels. So technically it can be spoofed - **so trust, but, verify**" and directs users to submit a problem report if they encounter AIS related errors [11, 12]. Additionally, the USCG maintains a Vessel Information Verification Service which is a website where you can find AIS static information discrepancies for vessels within the Nationwide AIS (NAIS) coverage [13]. Once again, the AIS system relies on the good faith of system users and vessel operators to manually input and verify that their data matches that on file with applicable governing agencies. This showcases the first major vulnerability of AIS, which is that users are inherently trusted to properly maintain their vessel's information which is broadcast without interruption to all surrounding stations within range. Even the presence of completely false data must be manually verified by an end-user using a website to submit a report. A recent collision between two towing vessels improperly displaying their static AIS data on the Mississippi River prompted the USCG to release a Marine Safety Alert titled "AIS – Accurate Broadcasts Don't Happen Automatically [14]". Since this collision involved the improper setting of the vessel's length between two vessels around a blind bend (i.e. initially not in visual sight of one another), other AIS system users should have been able to alert that the vessel was improperly displaying its AIS information and should remain clear. Therefore, relying solely on one vessel to determine its own information provides a single source of failure.

Due to its insecure design, AIS contains well-documented vulnerabilities that can easily be exploited by an adversary armed with a simple software-defined radio (SDR) and a VHF antenna and that could potentially cripple a major harbor. The vulnerabilities in the AIS system are a reflection of the time period in which it was created, and nearly two decades later we must implement solutions that adhere to the original design of the system as a public, navigation safety tool while ensuring bad actors cannot use this same data to cause harm to people or property. In 2018, members from the U.S. Coast Guard Research and Development Center (RDC), some of whom were involved in the initial creation of AIS in the nineties, stated that we must begin an international discussion of the requirements of "AIS 2.0" which should take into consideration national cybersecurity objectives [5]. Several other authors have researched AIS vulnerabilities and have suggested or developed solutions that would provide authentication and encryption to the entire AIS system. Additionally, several commercial and government products provide encrypted AIS transmissions for smaller subsets of vessels for use in law enforcement and other fleet activities where confidentiality is required. However, the void still remains for a secure public AIS system. The complete lack of security in the original design of AIS means that vessel data can be spoofed and hijacked. Additionally, AIS messages lack a time stamp and are therefore vulnerable to replay attacks. A bad actor can simply record a series of legitimate AIS transmissions from a vessel and replay them at any given time to create a fictitious target with real data. There is also no message integrity, meaning that there is no way to know if the data you received actually matches the data that was sent.

1.2 PROBLEM STATEMENT

The presence of these vulnerabilities within the decades old AIS system provide the motivation for this thesis. The goal of my research is to evaluate a feasible method to bring AIS up to the twenty-first century cybersecurity standards and eliminate the cause of two major AIS vulnerabilities: lack of source authentication and lack of message integrity. The major contributions I provide from this paper are twofold. First, I have built a Software Defined Radio (SDR) implementation of AIS which provides a robust test platform for system analysis. Second, I provide the details of a novel authentication protocol for securing AIS, based on the Timed Efficient Stream Loss-Tolerant Authentication (TESLA) protocol which enables receivers of multicast communications to authenticate the source and integrity of received data packets. Unlike the alternative approaches proposed for authentication in AIS, the approach presented here can authenticate messages without the use of an a priori shared secret key or the need to conduct key exchanges over several messages and requires significantly less overhead. This authentication protocol secures AIS by providing source authentication, ensures message integrity, and includes an explicit time stamp within data messages to prevent replay attacks.

1.3 THESIS OUTLINE

I will now provide a road map for the remaining sections of this thesis. Chapter 2 will cover the background information that is vital to analysis of AIS security. This will include a comprehensive review of the AIS system, including a technical overview using the Open Systems Interconnection (OSI) model. I will also review the fundamental elements of cryptography which must be understood before delving into the implementation of a security protocol for AIS. This chapter will conclude with a more detailed discussion of the existing vulnerabilities in the AIS system. Chapter 3 provides a review of the existing research that has been done on AIS security as well as an examination of research that is being done on similar problems in the aviation industry's Automatic Dependent Surveillance-Broadcast (ADS-B) system and Mobile Ad-Hoc Networks (MANET), which include block chain and Pretty Good Privacy (PGP) web-of-trust technology. Chapter 4 covers my first major contribution which is the implementation of an SDR AIS transmitter and receiver. Chapter 5 covers my second major contribution, namely the details of a novel authentication protocol for securing AIS, based on the Timed Efficient Stream Loss-Tolerant Authentication (TESLA) protocol. The work in Chapters 4 and 5 will be presented at the 2021 IEEE International Black Sea Conference on Communications and Networking. Finally, Chapter 6 will conclude the paper and identify several avenues for future work.

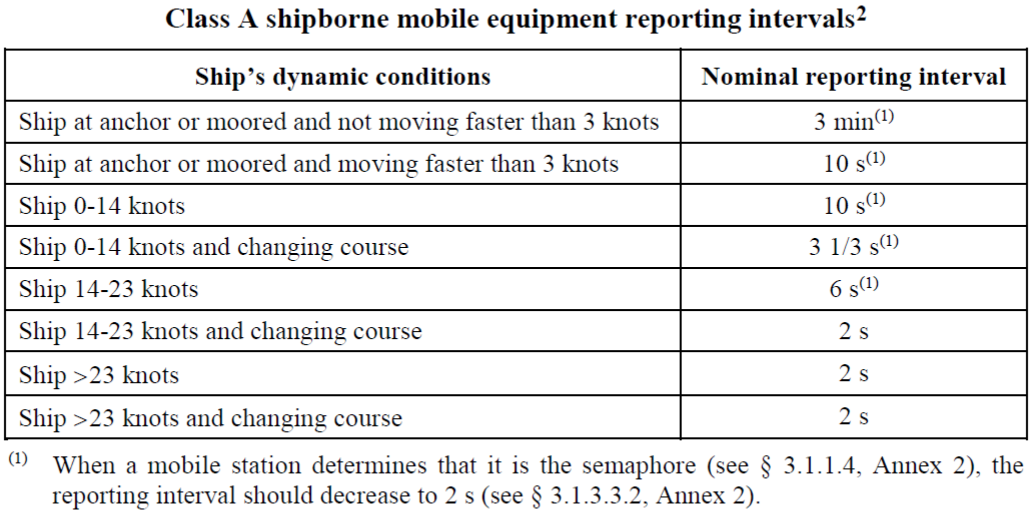
CHAPTER 2

BACKGROUND

This chapter will begin by providing a technical overview of the AIS using the OSI model and a breakdown of the Maritime Mobile Service Identifier (MMSI) used to uniquely identify every vessel. Next, there will be a brief overview of cryptography principles. Finally, a more detailed discussion of existing AIS vulnerabilities will be provided.

2.1 AIS TECHNICAL INFORMATION

AIS over-the-air transmissions are standardized by ITU M.1371-5 as a response to the IMO requirement for a universal shipborne AIS system to provide efficient communication between ships and shore stations. Internally, National Marine Electronics Association (NMEA) 0183 proprietary standard is used for data transmission between AIS and other electronic navigation systems at 4,800 baud [15]. Class A shipborne AIS systems comply with the IMO AIS carriage requirements while Class B devices are not necessarily in full compliance. Access to the VHF data link (VDL) should be accommodated through time division multiple access (TDMA) [1]. Self-organized time division multiple access (SOTDMA) is the preferred TDMA scheme for Class A devices since it appropriately accommodates users and makes efficient use of the radio spectrum. While many Class B devices use SOTDMA, some use carrier-sense TDMA (CSTDMA) which ensures the device only transmits when the network is free and does not interfere with SOTDMA Class A or B devices. Additional access schemes used by AIS will be discussed in Section 2.2.2. The system is designed to be autonomous, automatic, continuous and operate primarily in broadcast mode, although interrogation is possible [1]. Finally, AIS should be capable of expanding to accommodate future regulations which require more vessels to use the system. In general, AIS is a system which automatically and continually broadcasts a ship's dynamic and static information to all other stations in range and can receive and process the same information from others in a self-organized manner. Additionally, AIS is capable of transmitting safety related messages on request [1].



Class A shipborne mobile equipment reporting intervals[2]

Ship's dynamic conditions	Nominal reporting interval
Ship at anchor or moored and not moving faster than 3 knots	3 min[(1)]
Ship at anchor or moored and moving faster than 3 knots	10 s[(1)]
Ship 0-14 knots	10 s[(1)]
Ship 0-14 knots and changing course	3 1/3 s[(1)]
Ship 14-23 knots	6 s[(1)]
Ship 14-23 knots and changing course	2 s
Ship >23 knots	2 s
Ship >23 knots and changing course	2 s

[(1)] When a mobile station determines that it is the semaphore (see § 3.1.1.4, Annex 2), the reporting interval should decrease to 2 s (see § 3.1.3.3.2, Annex 2).

Fig. 2: Reporting Rate for Class A Device [1]

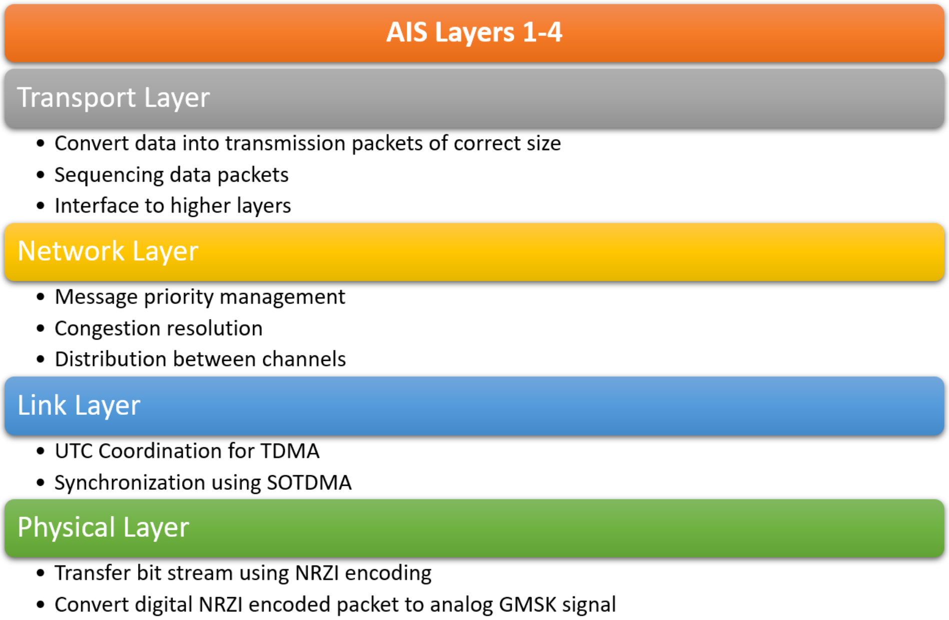
2.1.1 TRANSMISSION SCHEDULE

AIS will transmit a ship's static information (such as name, MMSI, call-sign, length) every six minutes, when data has been changed, or upon request. Dynamic information such as course and speed are much more pertinent to other vessels and thus this information is updated at a more frequent interval. The Reporting Rate (RR) for dynamic information is set at a variable rate based on a ship's navigation status, speed, and course. This means that if a vessel's status is listed as moored (not moving; attached to a dock or buoy) or at anchor (not moving; attached to the ocean floor), the AIS system will report at a less frequent interval than if the ship is listed as underway (moving). AIS is interfaced with the ship's global positioning system (GPS) which provides course over ground (COG) and speed over ground (SOG) data which are used to formulate the AIS reporting decisions. Fig. 2 shows the dynamic reporting conditions for a Class A AIS device.

Similarly, Class B devices have incremental reporting intervals based on speed, but they occur less frequently. Aids to navigation and AIS base stations also report at a set interval of 3 minutes and 10 seconds respectively.

2.2 ANALYSIS OF AIS USING OSI MODEL

In order to understand the AIS system design more clearly, I will conduct an analysis using the OSI model. AIS layers 1-4 and their general purpose are shown in Fig 3.



AIS Layers 1-4

Transport Layer

- Convert data into transmission packets of correct size
- Sequencing data packets
- Interface to higher layers

Network Layer

- Message priority management
- Congestion resolution
- Distribution between channels

Link Layer

- UTC Coordination for TDMA
- Synchronization using SOTDMA

Physical Layer

- Transfer bit stream using NRZI encoding
- Convert digital NRZI encoded packet to analog GMSK signal

Fig. 3: Layers 1-4 of AIS using OSI Model

2.2.1 PHYSICAL LAYER

AIS operates in the VHF maritime mobile band within the frequency range 156.025-162.025MHz. There are two primary channels, both with a bandwidth of 25kHz: 161.975MHz (AIS 1, default channel 1, 2087) and 162.025MHz (AIS 2, default channel 2, 2088). Additionally, maritime channels 75 and 76 are designated for long-range AIS usage with satellites. AIS encodes data at the physical layer using non-return to zero inverted (NRZI) encoding (change in level when a 0 occurs in bit stream) and Gaussian-filtered minimum shift keying (GMSK) modulation with a maximum time-bandwidth product of 0.4 at the transmitter and 0.5 at the receiver. The bit rate of the data should be 9,600 bit/s +/- 50ppm.

NRZI

This is a binary line code used for transmitting a binary signal to a physical signal, and in the case of AIS is used to transition binary data for transmission over the VHF channel. Data bits are denoted using the presence or absence of transition at a clock boundary. AIS denotes a transition as a 0, meaning that the presence of a clock transition denotes a 0 while the absence of a clock transition denotes a 1. Fig. 4 shows an example of NRZI with a transition associated with the symbol “0”.

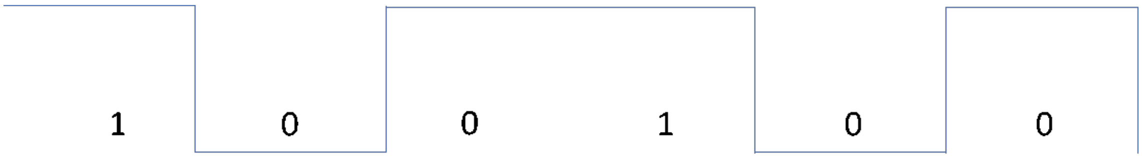


Fig. 4: NRZI Line Coding, Transition on 0

GMSK

This modulation scheme is used to transmit the NRZI line code over the VHF channel by varying the phase of the signal. GMSK is a form of MSK that applies a Gaussian filter before the signal is modulated. MSK uses a frequency separation of $(f_2 - f_1) = \frac{T_b}{2}$, where T_b is the bit period, and represents the minimum separation required for orthogonality using coherent detection and thus why the term "minimum" is used for this modulation scheme. MSK is a form of binary continuous phase frequency shift keying (CPFSK) which means that phase shifts are not as abrupt as they are in traditional FSK, which uses shifts in frequency to encode data and consequently results in drastic phase shifts. Fig. 5 shows a comparison between the phases of MSK and FSK transmission of a binary input.

These abrupt phase shifts in FSK are generated because the transmitter resets the frequency between each symbol, whereas a continuous phase modulation scheme has memory which allows phase transitions to occur based on the previously transmitted signal. A block diagram of the GMSK transmitter is shown in Fig. 6.

While Fig. 5 shows the difference between continuous and non-continuous phase modulation, GMSK provides the additional benefit of further smoothing phase transitions through pulse shaping prior to transmission. This can be seen in Fig. 7.

The discontinuities in phase jumps for a non-continuous phase modulation scheme such as FSK means that spectral efficiency is diminished, and therefore smoothing phase transitions through Gaussian filtering allows signals to be transmitted in band limited channels with greater ease. The effect on the spectrum can be seen through analysis of the power spectral density (PSD) of various modulation schemes. Fig. 8 shows a comparison of the PSD of MSK and Offset Quadrature Phase Shift Keying (OQPSK).

MSK falls in power much faster than OQPSK meaning it is more spectrally efficient. Since GMSK is pulse shaped using a Gaussian filter, sideband power is further reduced from regular MSK making it even more spectrally efficient and ideal for use in the maritime VHF spectrum [16]. Additionally, GMSK uses a constant envelope which makes demodulation of

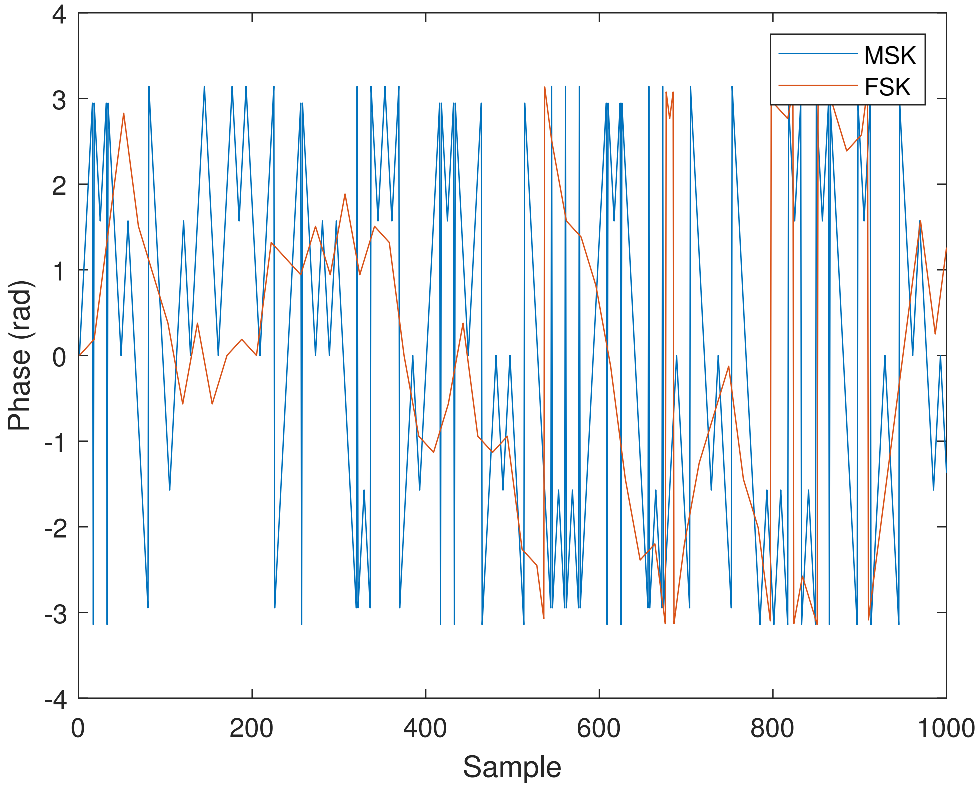


Fig. 5: Phase difference between CPFSK and FSK

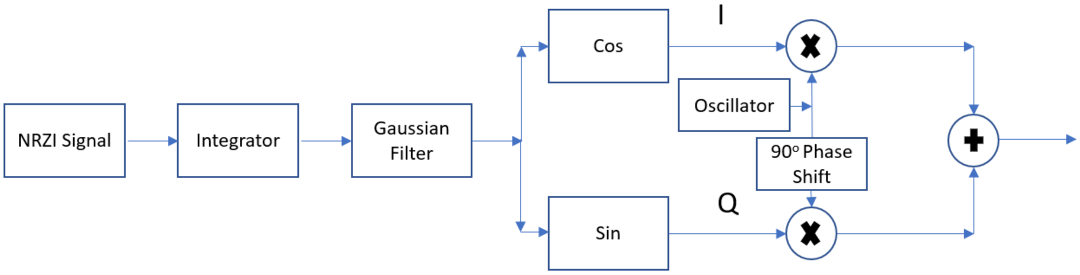


Fig. 6: GMSK Transmitter Block Diagram

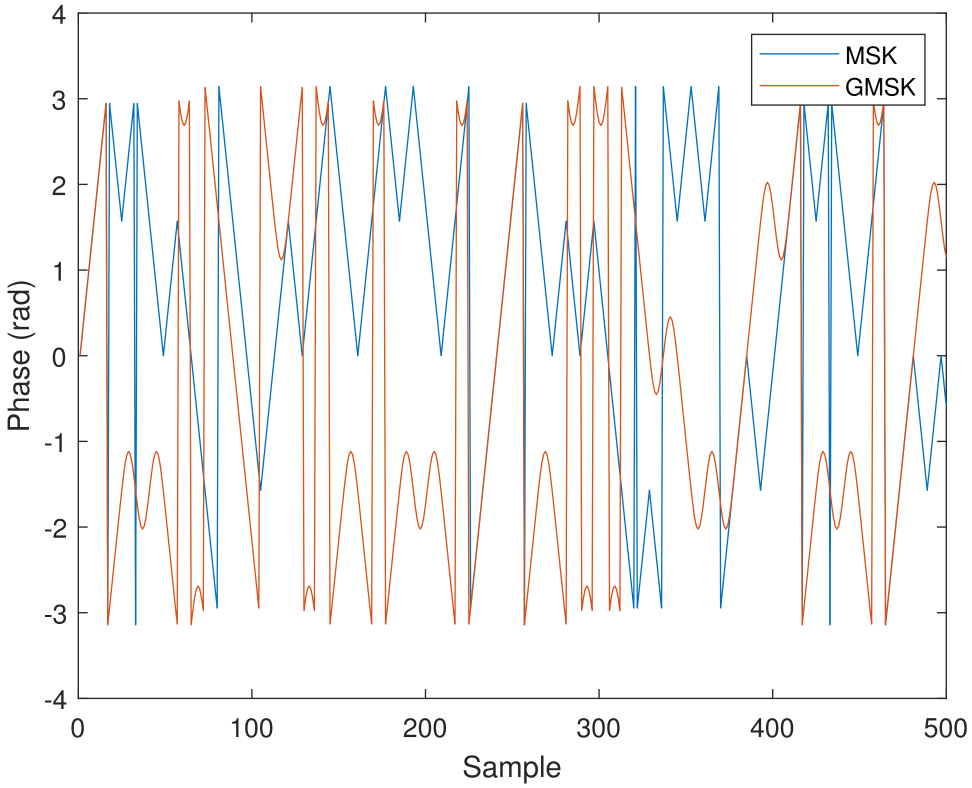


Fig. 7: Phase difference between MSK and GMSK

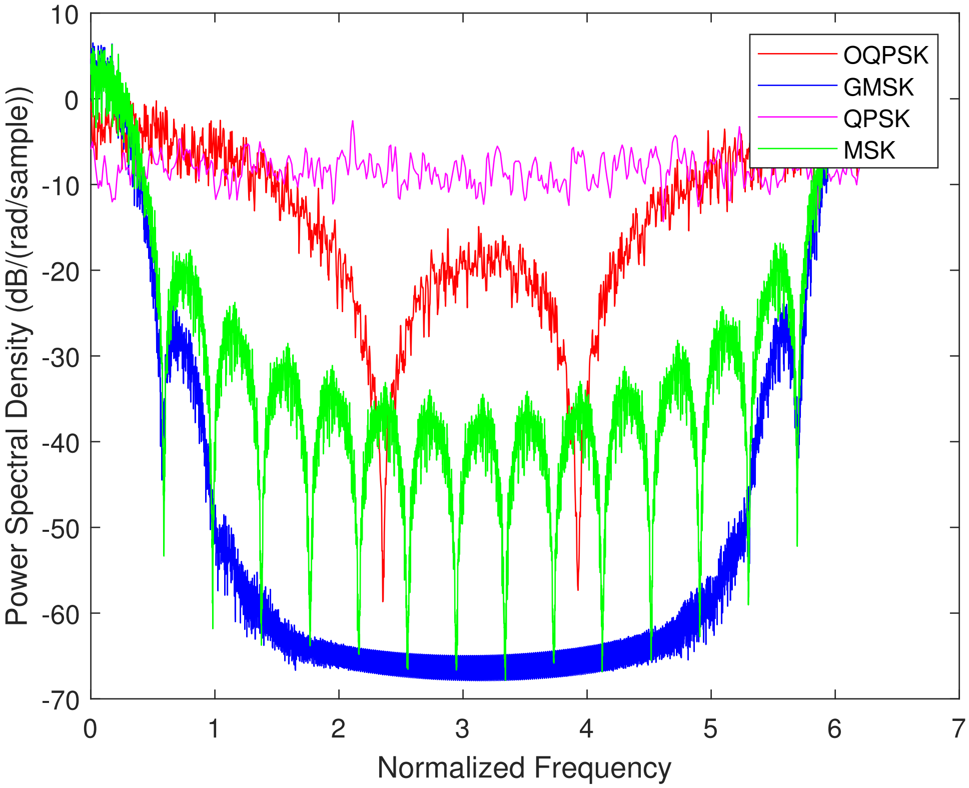


Fig. 8: PSD Comparison between OQPSK, MSK, QPSK, GMSK

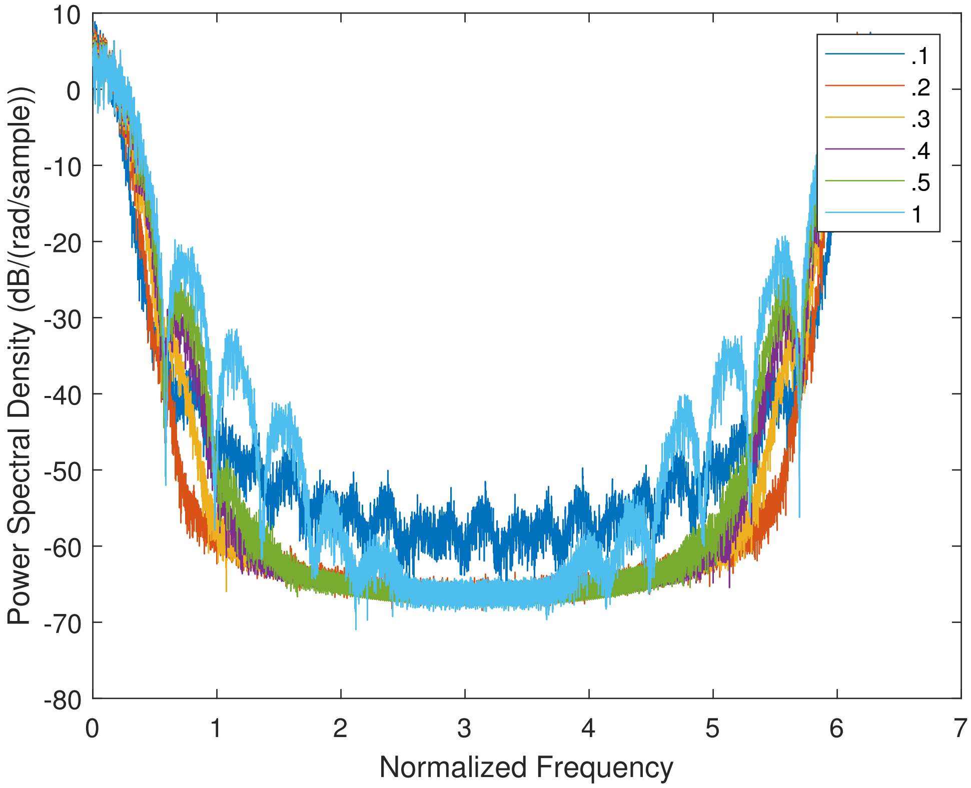


Fig. 9: BT Comparison for GMSK Signal

the signal fairly straightforward [17]. The time bandwidth (BT) product is used to further define the GMSK signal and is used to control the effects of the Gaussian filter. T represents the symbol period and B represents the 3dB (half-power) bandwidth, so the effect of the BT is to compress the signal within a smaller bandwidth space. Fig 9. shows a comparison of BT values for a GMSK signal with values ranging from .1-1.

The lower the BT value, the more spectrally efficient the signal is, but there is also a greater chance of inter-symbol interference (ISI) since the constellation points are closer together and thus the receiver will have a more difficult time correctly identifying symbols. For the AIS system, a BT between $.4 - .5$ is used which provides a good trade-off between detection at the receiver and bandwidth efficiency.

2.2.2 LINK LAYER

Data from the VHF channel is accessed using TDMA with a common time reference synchronized every 2 seconds for a mobile user and every 3.33 seconds for a base station. Users either have direct access to coordinated universal time (UTC) by setting a synchronization state to UTC direct, while other stations who cannot should synchronize their time off of nearby stations with the proper synchronization state set. Users cannot achieve indirect synchronization more than one user removed from UTC direct to avoid timing errors.

Frame

Data frames are one minute blocks of time divided into $2,250$ slots (indexed $0-2249$) with default access at the start of a frame. Frames are coordinated with UTC to start/stop with each UTC minute. This means that each slot is allocated $26.667ms$ for transmission. Users may begin transmitting Radio Frequency (RF) power at the start of a slot and must conclude within the allocated slots for transmission. Slots can be:

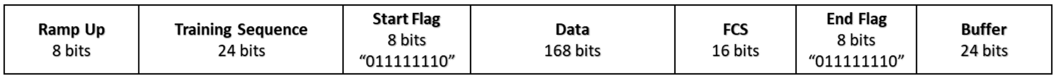
- Free - unused within receiving range
- Internal Allocation - allocated for transmission by own station; can be used for transmission
- External Allocation – allocated for transmission by another station
- Available - externally allocated by another station and is possible for reuse
- Unavailable – externally allocated by another station and cannot be a candidate for reuse

Data is transferred using high-level data link control (HDLC) specified by ISO/IEC 13229:2002 which includes a start and stop flag to indicate the presence of a data packet, and additionally includes a training sequence to synchronize the VHF receiver.

Data Packet Format

AIS data is transmitted in a packet consisting of 256 bits using the format shown in Fig. 10.

- Ramp up: Used between start of RF power and 80% RF power



Ramp Up 8 bits	**Training Sequence** 24 bits	**Start Flag** 8 bits “011111110”	**Data** 168 bits	**FCS** 16 bits	**End Flag** 8 bits “011111110”	**Buffer** 24 bits

Fig. 10: AIS Data Packet, Adapted from [1]

- Training Sequence: Sequence of alternating 0’s and 1’s and may begin with either a 0 or 1 since NRZI is used
- Start Flag: Standard HDLC flag used to detect the start of a transmission packet, set as 01111110 and is not subject to bit stuffing.
- Data: Contains the message being sent
- Frame Check Sequence (FCS): Cyclic redundancy check (CRC) to calculate checksum of data
- End Flag: Identical to the start flag
- Buffer: Allows for differentiation between messages from delay, sub-divided into the following:

  Bit stuffing – 4 bits

  Distance delay – 14 bits correcting for propagation delay of over 120NM (maximum possible is 235.9 nautical miles)

  Synchronization jitter – 6 bits used to preserve integrity of TDMA

Bit stuffing is utilized for data and the frame check sequence (FCS), which means that at the transmitter five consecutive ones should then have a zero inserted, while the receiver should remove the first zero after five consecutive ones.

A training sequence consisting of 24 bits alternating 0 and 1 is sent to synchronize with the receiver. Following the training sequence, the start flag is sent which is 8 bits and is defined as:

Start Flag: 011111110

Following 168 bits of data is a cyclic redundancy check (CRC) checksum based on the data portion of the frame to ensure the integrity of the data frame. Finally, the packet concludes with an end flag of identical construction to the start flag.

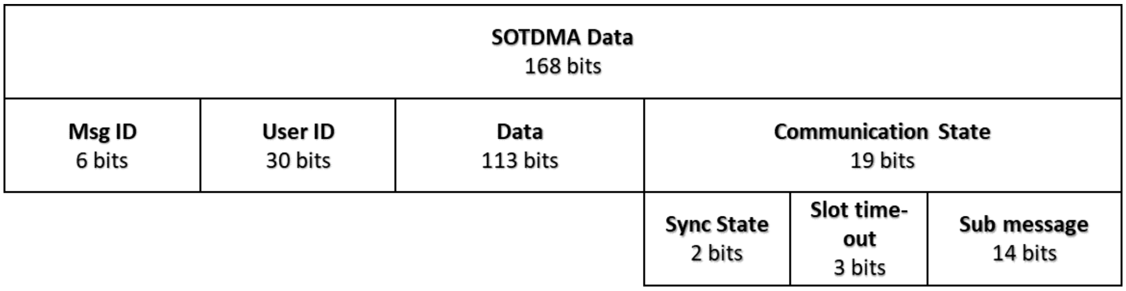
As stated previously, the 24 bit buffer space is used to account for differences in message lengths based on transmission effects and ensures messages are not transmitting over one another. Stations are allowed to occupy a maximum of 5 consecutive slots for continuous transmission and are only required to send a single set of overhead messages surrounding the data at the beginning/end of the transmission.

Access to Data Link

SOTDMA is the primary access scheme for the AIS system and is used mainly for repetitive messages on a scheduled interval from an autonomous station. There are three additional schemes for controlling data transfer when non-repetitive messages are required and when reporting intervals are changed. These access schemes are incremental time division multiple access (ITDMA), random access TDMA (RATDMA), and fixed access TDMA (FATDMA).

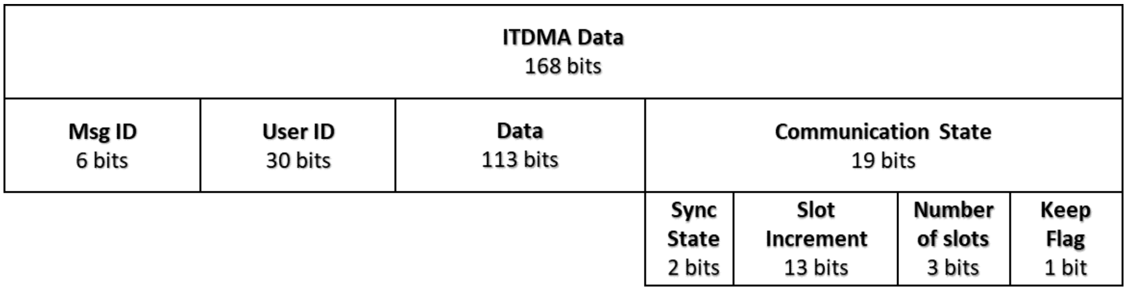
Upon entry into the network, the AIS device will monitor the VHF data link for 1 minute to determine a dynamic directory of all members and generate a frame map of the TDMA activity. After this initial elapsed time period, a user will enter the network entry phase where they wait for a nominal transmission slot (NTS) which is randomly selected among potential candidate slots within the selection interval using ITDMA to pre-designate a slot. Upon reaching the NTS, the user (if Class A mobile) will transmit a special position report (type 3) and then select its next NTS using the SOTDMA access scheme within the selection interval.

All messages contain a message ID within the data portion of the packet from Fig. 10, but the access scheme determines the remainder of the data structure. When using SOTDMA, the data portion of the packet is formulated as shown in Fig. 11 while ITDMA is formulated as shown in Fig. 12.



SOTDMA Data 168 bits					
Msg ID 6 bits	User ID 30 bits	Data 113 bits	Communication State 19 bits		
			Sync State 2 bits	Slot time-out 3 bits	Sub message 14 bits

Fig. 11: SOTDMA Data Structure, Adapted from [1]



ITDMA Data 168 bits						
Msg ID 6 bits	User ID 30 bits	Data 113 bits	Communication State 19 bits			
			Sync State 2 bits	Slot Increment 13 bits	Number of slots 3 bits	Keep Flag 1 bit

Fig. 12: ITDMA Data Structure, Adapted from [1]

2.2.3 NETWORK LAYER

The purpose of the network layer is to ensure messages are delivered in priority order, handle congestion resolution, and ensure messages are coordinated between the four possible AIS channels. As discussed in Section 2.2.1, AIS channel 1/2 are for ship-to-ship VHF AIS messages while channels 75 and 76 are reserved for long range satellite applications (Message 27). AIS is set to operate by receiving AIS channel 1/2 in parallel and transmit both periodic and non-periodic messages by alternating between channels every other message. For example, if initial link access and the first broadcast position report is sent on AIS channel 1, then the next periodic broadcast report will be sent on AIS channel 2. Responses to addressed messages should be conducted on the same frequency on which it was sent. AIS messages also contain four different priorities which aid in congestion control should messages require queuing. These priorities rank safety of navigation messages as the highest, while general information messages are lower.

- Priority 1: Position reports and link viability messages

- Priority 2: Safety related messages
- Priority 3: Assignment and interrogation messages
- Priority 4: All other messages

As Priority 1 messages deal with position reports and navigation safety, changes in RR resulting from a vessel altering its course, speed, or status may increase or decrease the number of these messages present in the link. Fig. 2 from Section 2.1.1 shows the dynamic schedule of possible RR based on vessel information. If a vessel increases its speed to a threshold that requires a change in RR, then the link layer ITDMA algorithm should be followed to identify a new NTS and then report at the new rate. Faster speeds mean vessels have less time to react, so it is of the utmost priority that these new messages are given access to the link. However, when a vessel decreases speed to a level that necessitates a new RR, the change should only occur after three minutes have elapsed at this new slower speed. This means that the AIS system leans on the side of caution and would rather have a slower unit reporting more rapidly than inadvertently miss a critical position report of a fast moving vessel. For course changes, a vessel's mean heading over the previous 30 second interval is compared to the present heading. Based on this information, a vessel is determined to be "changing course" if a heading change of greater than 5° is detected, and should be maintained until the change is less than 5° for 20 seconds. If the link becomes congested and therefore priority 1 messages may be in jeopardy of transmission, slots should be intentionally reused from distance stations ($>$ 120nm) in order to ensure that there are at least 4 candidate slots available for transmission. This ensures that the SOTDMA random selection of a transmission slot has at least 4 slots for use.

2.2.4 TRANSPORT LAYER

The final layer discussed here is the transport layer which ensures packets are appropriately sized and sequenced and should be the interface between the presentation layer. This layer has the important function of ensuring data packets are formatted properly so that they can be properly handled by other applications. For example, if a message requires too much data and exceeds the allowable number of slots for AIS transmission, it should reject the packet at the presentation layer. Additionally, this layer should correctly handle responses for addressed messages (Type 6/12) as well as ensuring broadcast messages are not acknowledged.

2.3 AIS MESSAGE TYPES

There are 63 possible AIS message types, with only message types 1-27 currently in use. These message types range from simple position reports, which provide an update to a vessel's GPS location, to aids-to-navigation reports that update the position of a buoy or navigation marker. A full list of these messages, along with their applicable priority, access schemes, and communication state can be found in [1].

2.4 MARITIME MOBILE SERVICE IDENTIFIER

The MMSI number is a unique nine-digit number issued to a vessel and is formatted as shown in Eq. 1. ITU-R M.585-8 standardizes the assignment and use of MMSI numbers throughout the world [18].

$$M_1 I_2 D_3 X_4 X_5 X_6 X_7 X_8 X_9 \tag{1}$$

The first three digits are the maritime identification digits (MID) and represents the country having jurisdiction over the vessel (the vessel's flag state). MID's are also assigned by the ITU and allow expansion if a country exceeds the number of unique six digit numbers following the MID. The U.S. has several MIDs including 338 and 366-369 [19]. The remaining digits $X_1 - X_9$ are a unique 9 digit identifier for the vessel.

MMSI numbers can also assigned to aids-to-navigation (ATON) such as buoys or lighthouses and would be of the following format:

$$\textit{Physical ATON: } 9_1 9_2 M_3 I_4 D_5 1_6 X_7 X_8 X_9$$

$$\textit{Virtual ATON: } 9_1 9_2 M_3 I_4 D_5 6_6 X_7 X_8 X_9$$

$$\textit{Mobile ATON: } 9_1 9_2 M_3 I_4 D_5 8_6 X_7 X_8 X_9$$

Aircraft can also be assigned MMSI numbers using the prefix 111. Search and rescue transponders (SART), man-overboard (MOB) and AIS-equipped emergency positioning indicating radio beacon (EPIRB) are considered emergency life-saving equipment and must have MMSI numbers assigned for identification of the owner of that specific device. These MMSI numbers are formatted as shown in Eq 2.

$$9_1 7_2 0_3 X_4 X_5 Y_6 Y_7 Y_8 Y_9 \tag{2}$$

$X_4 - X_5$ are the manufacturer ID, $01 - 99$, and $Y_6 - Y_9$ are sequence numbers $0000 - 9999$.

Similarly, Eq. 3 and Eq. 4 show the format for a MOB indicator and EPIRB device which each use a slightly different prefix but keep the same formatting scheme.

$$9_1 7_2 3_3 X_4 X_5 Y_6 Y_7 Y_8 Y_9 \tag{3}$$

$$9_1 7_2 4_3 X_4 X_5 Y_6 Y_7 Y_8 Y_9 \tag{4}$$

2.5 CRYPTOGRAPHY

The idea of secure communication predates wired and wireless communication and addresses the fundamental idea that most people or organizations are looking for when sending or receiving a message: how do I know what I received is unaltered, comes from who I expect, and is not viewed by anyone else? These questions can be briefly defined as the fundamental tenants of information security:

- **Confidentiality** – Only the sender and intended receiver should be able to understand the contents of the transmitted message, often referred to as encrypted or secure communications
- **Integrity** - Ensuring the contents of a message is unaltered
- **Authentication** – Confirming the sender and receiver are who they claim to be

The AIS system contains a simple CRC which only ensures there are no bit transmission errors; there is no confidentiality, integrity, or authentication. As discussed in section 2.6, the system has many holes which could benefit from the application of basic security principles. One of the simplest methods of providing encryption is to take a plain text message and apply a key, K_A, to the message which encodes the data using an algorithm. Now the text is unreadable to a human and requires a key to undo the encryption of the message and read its contents. Based on this principle, several encryption methods exist [20].

2.5.1 SYMMETRIC (SECRET) KEY CRYPTOGRAPHY

In symmetric key cryptography, the same key (called a key pair) is used for both encryption and decryption. In this example, m is the plain-text contents of a message and K_A is an encryption key used by “Alice” to seal the contents of the message so that nobody else can read it. However, if Alice gives this key to her friend “Bob”, he can use K_A to decrypt the message. This means that the following equation holds true:

$$m = K_A[K_A(m)]. \tag{5}$$

This form of encryption is one of the oldest, most classical ways of conducting secure communications between users. However, the issue here lies in the fact that the secret key must remain secret, which itself requires a secure method of transmitting the key. Without knowing the secret key, you would be unable to read an encrypted message, yet you cannot encrypt a message without knowing a secret key, so therein lies a conundrum. Users can physically exchange keys, but this limits the scope of the encryption mechanism and if the key is somehow compromised, then all messages between the users could easily be intercepted, viewed and altered. Additionally, symmetric key cryptography does not provide authentication since keys are identical and must be shared. One method of symmetric key cryptography currently in use is the Advanced Encryption Standard (AES) which uses 128, 192, or 256 bit keys and has been proven to be resistant to brute force attacks [20].

2.5.2 ASYMMETRIC (PUBLIC) KEY CRYPTOGRAPHY

A Public Key Infrastructure (PKI) can be created to eliminate the need to share a secret key as seen in the symmetric key scenario. Instead, Alice and Bob can communicate with their own private key (known only to themselves) and share a public key (known to everyone). In order to be sure the public key is valid, it is signed by a Certificate Authority (CA) which holds the sole job of validating and issuing certificates that bind the key with the identity. This CA role can be managed in a variety of ways but should be a highly trusted third party (TTP) contact for validation of public keys. If the CA is untrustworthy or becomes compromised, the validity of every user in the PKI structure comes into question which makes this a significant point of failure [20].

In this scenario, the keys will be denoted as follows:

Alice’s Private Key: K_A^-

Alice’s Public Key: K_A^+

Bob’s Private Key: K_B^-

Bob’s Public Key: K_B^+

Using these four keys, Alice and Bob can exchange a plaintext message without the need to exchange a secret key. This occurs because the key pairs are generated in such a manner that one of the keys is the only possible method to decode a message encrypted with the other. This means that the following equations hold true:

$$m = K_A^-[K_A^+(m)] = K_A^+[K_A^-(m)] \tag{6}$$

$$m = K_B^-[K_B^+(m)] = K_B^+[K_B^-(m)] \tag{7}$$

These types of public/private key pairs can be generated using the Rivest, Shamir, Adelson (RSA) algorithm which uses the following steps:

- Select two large prime numbers, p, q
- $n = pq, z = (p-1) * (q-1)$
- Select $e < n$, such that e has no common factors with z
- Select d such that $ed * mod(z) = 1$
- Public Key: (n, e)
- Private Key: (n, d)

To encrypt a message m, compute the following:

$$c = m^e * mod(n) \tag{8}$$

To decrypt a message, c, compute the following:

$$m = c^d * mod(n) \tag{9}$$

The fact that either the public or private key can decrypt the other as shown in Eq. 6 and Eq. 7 can be proven using Eq. 9 and modular math properties.

$$c^d * mod(n) = m$$

$$c = m^e * mod(n)$$

$$n = pq$$

$$z = (p - 1) * (q - 1)$$

$$c^d * mod(n) = (m^e * mod(n))^d * mod(n)$$

$$c^d * mod(n) = m^{e*d} * mod(n)$$

$$(m^e * mod(n))^d * mod(n) = m^{e*d} * mod(n)$$

$$(m^e * mod(n))^d * mod(n) = (m^d * mod(n))^e * mod(n)$$

Therefore, if Alice wants to send a message to Bob that only he can read, she can encrypt the message using his public key, and as long as his private key remains known only to him, Bob will be the only one who will be able to decrypt the message as shown in Eq. 7.

Using a similar principle, Alice and Bob can use these four keys to create a digital signature by reversing the order in which the keys are used. Instead of using Bob's public key to encrypt a message, she can encrypt a message using her own private key (known only to her) and when Bob receives the encrypted message, he can use Alice's public key to decrypt the message which shows that only Alice could have sent the message. However, this does not provide confidentiality since Alice's public key is known to everyone.

One potential issue that arises with PKI is that digital signatures using public key encryption mechanisms require 3-5 orders of magnitude more processing power than symmetric key cryptography. To reduce the size of a message's contents, a hashed message authentication code (HMAC) can be used. First, the entire message and a secret key are sent through a hash function, which essentially provides a one-way, fixed size fingerprint of the message that cannot be reversed. The HMAC is then sent as an attachment to the original plain text message. The receiver will then compute the hash of the plain text message along with the secret key and compare this to the HMAC that was sent. If they are identical, the receiver can be assured the message is authentic and has not been changed in transit. However, the HMAC must necessarily include an initial, lengthy symmetric key exchange using PKI to ensure that only the sender and receiver have access to the key used in the HMAC; otherwise, there would be no way to verify the authenticity of the sender. Therefore, the HMAC reduces the computation and overhead of individual messages, but requires a several-message exchange with each user at the beginning of a transmission in order to agree upon a key.

2.5.3 IDENTITY BASED ENCRYPTION

In 1985, Adi Shamir built upon the generic public key cryptography scheme and proposed a methodology called Identity-Based Encryption (IBE) which removes the need to store and retrieve another user's public key from a database or repository [21]. In his system, he proposed that individuals are given a smart-card by a trusted key generation center. The card contains their private key as well as the ability to decrypt all other user's public keys. The novel concept here is that the other users' public keys are generated based on other publicly available, unique information such as a name or e-mail address. This means that users only have one interaction with the trusted third party to obtain their smart card and no longer have to conduct frequent exchanges to verify the public key of every user. The obvious issue in this implementation is that the trusted third party contains the mechanism to generate private keys for every individual and thus those secrets must be closely guarded. If compromised, all users in the system will need to be re-issued new keys. Additionally, the RSA scheme discussed earlier is unable to meet the requirements of this system due to the fact that a seed value must be able to generate private keys for all users and that same seed value must not be computable from the public/private keys. In Shamir's proposal, the signature can be generated using the following equation:

$$s^e = i * t^{f(t,m)} * mod(n) \tag{10}$$

The function f is chosen by the trusted third party as well as common values of n and e, but the trusted third party is the only one who knows the factorization of n. The user's identity, i, is unique and public, and the private key generated by the trusted third party takes the form:

$$g^e = i * mod(n) \tag{11}$$

A user can sign a message using the equation:

$$t = r^e * mod(n) \tag{12}$$

And verification is conducted using:

$$s = g * r^{f(t,m)} * mod(n) \tag{13}$$

Shamir recommended extending each user's identity into a pseudo-random string in order to improve security and reduce relationships between identities. This would not change the identity basis of the scheme since all users would know the pseudo-random generation function and be able to compute the pseudo-random string of any other user's identity. While Shamir's scheme does not eliminate the need for a trusted third party altogether, it does alleviate some of the burden associated with public key exchanges while maintaining authentication.

2.5.4 BLOCKCHAIN TECHNOLOGY

In recent years, the blockchain has become a prominent feature within the financial community to provide anonymous exchange of digital currency, but its benefits have extended into many communities. Many researchers are working to adapt this technology and apply it to their fields, including secure communications. NASA has recently looked into applying a blockchain-based protocol based on IBM's Hyperledger fabric to overhaul ADS-B security, so there is potential that a similar approach can be used for AIS [22]. The original idea for the blockchain was published by Satoshi Nakamoto in [2] and contained the basis for a peer-to-peer electronic cash system that eliminated the need for a third party financial institution. Instead, Nakamoto conceptualized a hash-based proof-of-work to create an immutable record of all previous transactions so long as the collective CPU power of the honest users is greater than that of would-be attackers. In this paper, the goal is to figure out a way to prevent "double-spending" which requires knowledge of previous transactions and balances. This is traditionally accomplished through the use of a trusted-third-party (TTP) who receives and validates every transaction. Eliminating this TTP means that every node needs to have knowledge of every previous transaction. Essentially, there needs to be a mechanism for all nodes in the network to collectively agree upon one common history of transactions. The first step to achieving such a mechanism is the creation of a time stamp server that can prove data existed at the time stated by creating a time-value hash chain. Next, a proof-of-work (POW) is required to implement this time stamp server within the network. The POW is a complex computation involving an incremental nonce that requires exponentially more CPU power for each instance generated. The result of the POW is a value that can be used in a chain to provide a record of all previous work, and any changes would require redoing each computation. This POW also allows for a fair decision making process and includes a time-based mechanism to slow down calculations if they are being computed too quickly which could lead to vulnerabilities. Within this framework, all transactions are broadcast and stored as blocks, where the longest chain is considered to be the correct object. To save space, Merkle Trees are used for storage which means only the

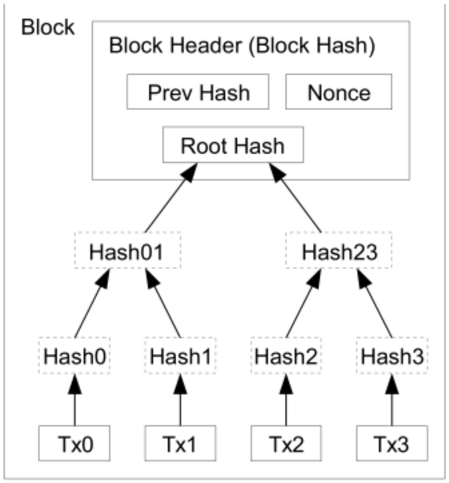


Fig. 13: Binary Merkle Tree Example, Derived from [2]

root must be included in the hash function and branches can be pruned. A Merkle Tree is a method to simplify the storage of hash functions by concatenating a number of child nodes into parent nodes as successive branches of a tree, ultimately resulting in a root node derived from all nodes beneath it. An example of a binary Merkle Tree can be seen in Fig. 13.

From Fig. 13, Hash0 is created by taking the hash of Tx0 and Hash1 is created by taking the hash of Tx1. Hash0 and Hash1 are then concatenated, and then the hash of that is taken to produce Hash01. Moving up the tree, if any given user can verify the root hash is valid, it proves that all children nodes produced from this root are also valid. In Bitcoin and other blockchain based payment systems, users only need to keep block headers (which contain the Merkle root) for the longest existing POW chain, and the accuracy lies in the fact that it is a valid part of the blockchain.

Since blocks are generated using complex computations, there is a possibility that an attacker generates the next block in the chain. However, since the blocks are built upon one another, this would also require the attacker to derive each previous block in the chain; thus, the problem becomes exceedingly complex as the length of the blockchain increases. Assuming the attacker starts from behind, the probability to catchup, q_z, can be calculated as follows, where q is the probability the attacker finds the next block, p is the probability a valid user finds the next block, and z is the number of blocks the attacker is behind the valid user:

$$q_z = \begin{cases} 1, & \text{if } p \leq q \\ (\frac{q}{p})^z, & \text{if } p > q \end{cases}$$

You can see that as z gets larger, the probability that the attacker will ever catch up grows exponentially smaller. Therefore, once the blockchain is initiated, every successful addition makes it increasingly more secure thus creating an immutable chain that can maintain a public record of secure transactions from numerous users.

There is potential to use a blockchain based system to conduct secure maritime communications or conduct PKI exchanges in order to eliminate the need for a CA.

2.5.5 PGP WEB OF TRUST

PGP was developed by Phillip Zimmerman in the early 1990s during a time when e-mail was just beginning to come online. He wanted to develop a system that "...empowers people to take privacy into their own hands" instead of relying on government controlled encryption protocols [23]. The foundation of PGP is a traditional PKI system where users generate public and private keys used for encryption and digital signatures as described in Section 2.5.2. However, Zimmerman delves into the methodology for how "trust" in a public key is established. He raises the dilemma where a user wants to send an encrypted message to Alice and retrieves her public key from a database, but unbeknownst to the sender, this public key actually belongs to Charlie who has generated a public key with Alice's identity. When the user uses this public key to encrypt a message, Charlie will actually receive the message and read its contents, not Alice. Typical PKI structures use a TTP to sign public keys and bind them to a specific user to prevent this problem from occurring. PGP expands upon this notion and allows individual users to sign public keys by incorporating the notion of an "introducer" who signs a copy of Alice's public key, essentially vouching for its authenticity. For example, if you know David is a trustworthy source, and if David has signed Alice's public key, you can verify David's signature on Alice's key and be assured the key actually belongs to Alice.

The author notes that protecting public keys and ensuring they are associated with the correct individual is the most difficult problem in PKI but believes that the social dynamic of individuals interacting and signing each other's keys is the natural way to handle this dilemma. As individuals sign public keys and post them to a centralized database, a web of trust is formed where links in the web are established via trust sources that exist between individuals. As this web grows larger, one will inevitably find a path to establish a trust

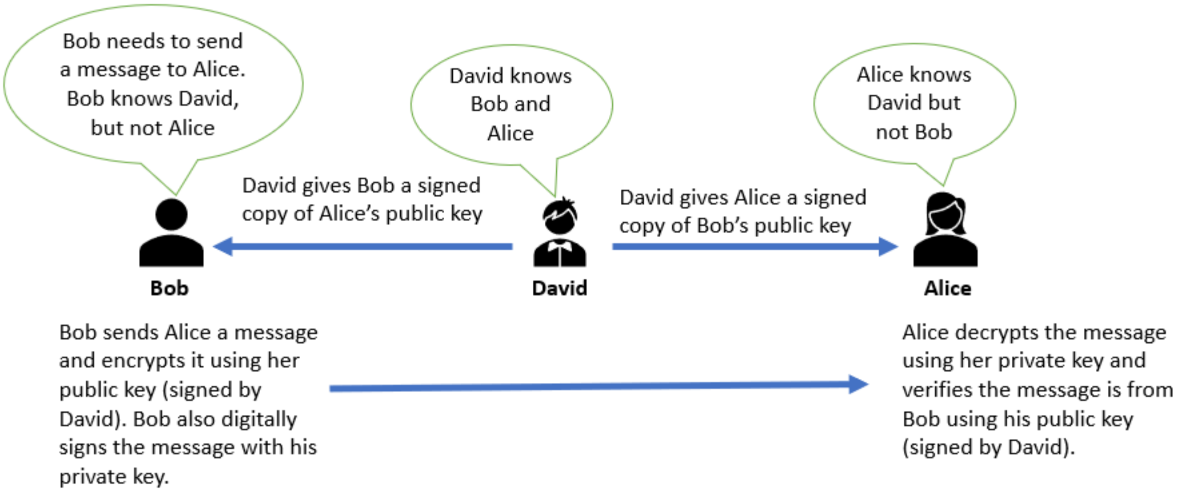

Bob needs to send a message to Alice. Bob knows David, but not Alice
David knows Bob and Alice
Alice knows David but not Bob
David gives Bob a signed copy of Alice's public key
David gives Alice a signed copy of Bob's public key
Bob
David
Alice
Bob sends Alice a message and encrypts it using her public key (signed by David). Bob also digitally signs the message with his private key.
Alice decrypts the message using her private key and verifies the message is from Bob using his public key (signed by David).

Fig. 14: PGP trust model of public key verification using introductions

link between themselves and another individual so that minimal introductions are needed. It is also important to distinguish between trust in the integrity of who the source claims to be, and trust in the individual. For example, just because David trusts that Alice's public key belongs to her does not mean that David trusts Alice as an individual; David simply knows with certainty that her public key is verifiable. Users in PGP can also choose who they want to trust. There is no requirement that one must accept a key through an existing trust relationship. For example, just because Bob trusts David and David trusts Alice, Bob does not necessarily have to trust keys from Alice. In [24], a novel upgrade to PGP was proposed that added in a feedback element where a user can negatively sign a public key, essentially stating that they do not believe the public key belongs to who it claims. Their evaluations showed that even with untrustworthy or malicious users, their model of trust allowed for better coverage and accuracy of the web of trust when extended to many users. Ultimately, PGP provides an alternative approach to the handling of public key signatures than the traditional root CA.

2.6 AIS VULNERABILITIES

Chapter 1 introduced the fact that AIS was designed as an inherently insecure system. To specifically address the ways in which these vulnerabilities can be exploited, the authors in [25] pointed out the following security issues within the AIS system:

1. Ship spoofing – Assigning static and dynamic AIS information to a fake ship and

planting that vessel in any location in the world.

2. ATON Spoofing – Similar to ship spoofing, this attack involves creating a fake ATON beacon and placing it in a false location within harbors to direct vessels into danger

3. Collision Spoofing – Since AIS was created to reduce risk of collision, this feature can set off alarms using their first identified threat (ship spoofing) by placing a fictitious vessel on a collision course with another real vessel.

4. AIS-SART Spoofing – Similar to ship spoofing, this attack includes creating a fake AIS-SART beacon that would lure rescue forces into a specific area in order to assist with the distress.

5. Weather Forecasting – Using binary messages to convey false weather alerts.

6. AIS Hijacking – Modifying a real user's AIS static or dynamic information to falsify the vessel's location, name, speed, or flag state.

7. Availability Disruption – Impersonating maritime authority using existing AIS messages to disable all AIS communications within a large geographic area.

8. Frequency Hopping - Impersonating a maritime authority to force users to change their AIS frequency, rendering the system useless since the user will have nobody to transmit/receive information with. AIS is designed to have such a command persist even after a reboot.

9. Timing Attack – An AIS transponder is instructed to delay its transmission or transmit at an extremely fast pace and overload the SOTDMA process of AIS messages amongst users.

2.6.1 AIS MESSAGE 21: AID-TO-NAVIGATION (ATON) REPORT

This message is sent out to provide the position and status of ATON but due to the lack of AIS authentication can be manipulated or falsified. Although physical ATON still exist in harbors to guide ships into and out of port, electronic ATON can be implemented by a competent authority to mark the virtual position of where an ATON should be located if it was either moved off station due to weather or damage. While this technology is greatly beneficial to mariners and the USCG alike, the potential for harm is possible. In shallow water ports with narrow channels such as the Chesapeake Bay's Thimble Shoals channel, an adversary would simply need to relocate electronic AIS beacons a few hundred feet north or south, which would potentially lead mariners to falsely believe they are out of the channel and potentially run aground. Aside from the environmental and logistical challenges associated with such a disaster in a narrow channel, this could also lead to other defense and homeland security related issues since a major U.S. Navy maritime port with numerous nuclear-powered vessels is now blocked for entrance/exit.

2.6.2 AIS MESSAGE 22/23: CHANNEL MANAGEMENT/GROUP ASSIGNMENT COMMAND

These messages are designed to be used by competent authorities to set AIS VHF and operational parameters directed at either a specific vessel or region. For example, an AIS message 22 can be sent to a specific vessel using its MMSI number and direct that vessel's AIS transceiver to shift from the traditional AIS frequency to an alternately designated frequency. Similarly, this same message can be directed to all vessels within a geographic area and cause their units to shift AIS transmission channels. Message 23 can also direct a vessel's or group of vessel's AIS transceiver into a maximum 15-minute quiet time. These message types are very dangerous since an adversary can force a vessel to stop broadcasting and/or receiving AIS data without the vessel even noticing. Even if a vessel did notice the message, an automated series of commands could be run in quick succession to shift users to various, random channels, leading to a perceived AIS outage in a particular area. In 2010, the USCG was conducting NAIS testing and broadcast an AIS message 22 and directed vessels between Connecticut and North Carolina to shift their broadcasts to non-standard AIS frequencies, essentially forcing them to become silent to other users and also lose reception of appropriately tuned AIS users. The USCG stated that "the channel management information will stay in memory for 5 weeks or until an affected vessel moves more than 500 nautical miles from the defined region. AIS channel management commands can only be manually overridden or erased by the user via the unit's channel management function or automatically overridden via another channel management message for the same defined region. Re-initializing or resetting your AIS or transmission channels will not necessarily reprogram your unit back to the default channels [26]." These messages clearly serve an important purpose from a national security or law enforcement perspective, but there must be tighter control over their source.

It is clear that AIS lacks two fundamental elements of a secure communication system:

user authentication and message integrity. The majority of AIS vulnerabilities described thus far exist because there is no way to verify who is sending a message; thus, all receivers act blindly in response to any message. Additionally, messages contain no time stamps which means they can be falsified or replayed at any time. MMSI numbers are required for every AIS transmission and due to their unique nature, cannot be reused by more than one vessel throughout the world. However, anyone can simply broadcast a false MMSI number of their choosing since there is no check to verify whether it belongs to the registered vessel. This also causes another issue due to the way AIS messages are handled locally among a group of vessels. If an adversary spoofs an AIS target using the MMSI number of another vessel within VHF range, the system will cause the target to “jump” around each time the vessel’s position is broadcast. Essentially, an ill-minded actor can “move” the position of a real AIS target causing confusion to other vessels in the area. In order to validate that many of these vulnerabilities exist, I used a simple Software Defined Radio (SDR) setup to generate, transmit, and receive AIS messages. In Chapter 4, I will describe the background of SDR and how it can be used to analyze signals and recreate these attacks.

CHAPTER 3

RELATED WORK

Despite the existing research that has been conducted on the AIS system, there remains a very limited set of work that specifically addresses the technical changes needed to implement security features in the AIS system. Several authors have discussed AIS from a policy perspective and have offered potential validity checking solutions that are typically shore based in nature and therefore do not adequately represent realistic solutions to conduct on-the-fly authentication and message integrity checks during random vessel encounters at sea [27]. In this chapter, I will explore the research that has been conducted on AIS security and will also look into the work being done in the aviation and vehicular security fields, as they face similar challenges.

3.1 AIS SECURITY

In 2014 the authors in [25] conducted what appears to be the first comprehensive security evaluation and verification of vulnerabilities within the AIS system. Researchers at Trend Micro Research used SDR to create an AIS transmitter called “AISTX” in GNU Radio which allowed them to manipulate AIS frame data and test their vulnerabilities. Using their SDR, they aimed to find the various ways AIS is vulnerable. First, the authors took a software based approach to conduct spoofing and man-in-the-middle attacks by providing fake vessel data to a popular website MarineTraffic.com. This vulnerability does not necessarily show a weakness in the AIS system itself but highlights the websites that profit off of the aggregation and sharing of plain-text, publicly available AIS information. The authors’ major contribution came from their creation of an AIS Frame Builder block for use in GNU Radio and their validation of significant holes in the AIS framework using simple SDR tool kits.

To overcome the vulnerabilities listed in Chapter 2, the authors recommended implementation of anomaly detection techniques by VTS or other competent authorities to detect and flag suspicious activities. The key problems here are identifying which parameters would need to be set and how to make them detailed enough to detect issues without overwhelming the system. Additionally, the ocean is a vast international space and identifying an entity to oversee offshore issues would likely be logistically impractical. Additionally, the authors recommended installing X.509 PKI infrastructure to use digital certificates issues by a competent authority to validate AIS users. Digital signatures are a great method to verify the authenticity of a message, but X.509 certificates carry significant data overhead and are not feasible for use in the band-limited AIS spectrum [28]. The logistics and architecture of a global PKI system will also be quite complicated. Additionally, PKI requires database access to retrieve the public key of any vessel that is encountered, which means either downloading and keeping a local copy of every ship in the world's key, or using an internet-based lookup service to download keys on-the-fly. Many vessels have limited or non-existent internet access while operating offshore which makes these solutions difficult and would require operators to consistently sync their devices while connected to the internet. Additionally, it is difficult to identify one central competent authority who can issue certificates to work in international waters, waters which are legally not regulated by any one country.

In [29], the authors found and validated similar vulnerabilities to [25] but opted to use simulated GPS data fed into a standard AIS transmitter instead of SDR. The authors proposed an IEEE 1609 Influenced AIS system where a competent central authority issues a certificate that validates un-alterable static AIS information on a vessel's unit. They proposed a three tier system that allows increasing levels of access to information about vessels around them. For example, tier one is labeled as "navigational safety mode" and only transmits a vessel's location, course, and speed while protecting private information (which they identify as everything except positional data). Tier two allows for encrypted exchange of information between vessels and requires users to accept or reject access to vessel requests for information. Finally, their third tier is reserved for security organizations and allows them to access any information about another vessel without requiring their consent. This solution suffers from the same competent central authority problem as proposed in [25], as it is based on one trusted central authority granting users private keys. Additionally, this solution removes many of the most important features about AIS, which includes obtaining the name, destination, and call sign of another vessel. This information is not private (in fact, it is required to be written explicitly on the vessel itself) and is extremely useful to mariners and aids them in conducting proper radio calls and navigating busy waterways where multiple vessels are located. Although the collision avoidance elements of AIS gained through positional (GPS) data are important, the situational awareness tools are something that cannot be removed from the system. Additionally, requiring users to manually accept/reject requests for AIS information is both cumbersome and unrealistic for a mariner navigating a congested waterway. Transiting through areas such as the Panama Canal, Strait of Malacca, or even New York Harbor often includes hundreds or even thousands of AIS targets, and manually or automatically sorting this data would not be possible and could even be distracting and dangerous.

[28] built upon the recommendation from [25] to implement SecureAIS, which is a software-only method to provide encryption and authentication using a pairwise key developed within the bandwidth constraints of the existing AIS infrastructure. Their methodology uses Elliptic Curve Qu-Vanstone (ECQV) implicit certification scheme and Elliptic Curve Diffie-Hellman (ECDH) key agreement algorithm. The ECQV implicit certification differs from a traditional certificate in that it can be extracted by a third-party using the implicit certificate of a CA (in the form of a cryptographic value) and identification data of another user (such as their MMSI). In order to generate an individual user's prublic/private keys using the ECQV scheme, the following variables must be defined. First, the elliptic curve group, ϵ and a generator, *G*, as well as a hash function, *H(-)* are all required to ensure functions are of the same format. The CA has a public key, *C*, known to all users and a private key, *c*. Requesting the implicit certificate, *M*, is done by a user with ID I by first generating a pseudo-random number, *n*. Next, $N = n * G$ is computed and sent to the CA who also generates their own pseudo-random number, *k*. The CA can then compute the implicit certificate using Eq. 14 and the implicit signature, *s*, using Eq. 15.

$$M = N + k * G \tag{14}$$

$$s = c + k * H(M, I) \tag{15}$$

These are then sent to the user who can verify their authenticity and then finally computes its private key using Eq. 16 and Eq. 17.

$$p = s + n * H(M, I) \tag{16}$$

$$P = p * G \tag{17}$$

Finally, using this information it is possible for any user who knows M (the implicit certificate) to generate another user's public key using Eq. 18 while only knowing their identity.

$$P = C + H(M, I) * M \tag{18}$$

These public key certificates can be extracted faster and require less data transfer than a traditional X.509 certificate as proposed by the authors in [25] which did not consider the size of the existing AIS data packets. To implement this system in the AIS framework, the authors note that a two phase process is required. The first phase is the "setup phase" which is done every time the AIS transceiver is turned on and the public and private keys are generated using the previous equations above. The role of the CA is played by a central maritime authority such as the IMO and the identity information of each vessel is a combination of the MMSI number and the expiration date of the cryptography provided by the CA. The second phase called the "online phase" occurs when two vessels interact and need to share information, which is done using AIS binary messages (Type 6). This is done through an ECDH scheme and involves a series of transmissions back and forth between two vessels where they share cryptological information. First, a randomly generated nonce, M_A, and security level indicator (specifying desired security level) is sent from user A to user B. User B will validate the information received, check whether the materials provided are not expired, and verify that it can locally support the security level indicated by user A. User B stores the nonce and then uses Eq. 18 to generate the public key of user A. Finally, user B generates a temporary session key that is sent back to user A along with M_B generated in the setup phase and a randomly generated nonce. User A performs the same functions as user B to generate the public key of user B, and then generates an authentication proof using the temporary session key which it will send to user B so that they can verify that only user A could have sent the message. User B can now use the two nonces generated during the exchange to generate a final session key that is sent to user A who computes the same process, resulting in a mutually agreed upon key to be used for symmetric key cryptography between the two users. As long as the certificates remain valid, this exchange only needs to happen the first time two users interact and subsequent meetings can use the same session key. The authors used the software ProVerif to verify that their protocol is secure against man-in-the-middle and replay attacks. They also verified their protocol using a X310 SDR and completed several experiments to examine data and time usage to establish their secure connections. First, using the largest, 256 bit, security level required 20 time slots to establish a shared key compared to 96 time slots for an X.509, a 79% reduction in overhead. Using 80-bit security requires only 10 time slots with secureAIS, 20% reduction in overhead from 50 time slots required with X.509 certificates suggested by [25].

However, this method suffers from several issues. First, even at the lowest level of security users must exchange 10 total messages to conduct sender/receiver authentication and establish a session key. Given that each frame is 2250 time slots, this means that only 225 simultaneous pairs of users (450 total) would be able to authenticate with each other at one time in one geographic area before overwhelming the TDMA scheme. This also opens up a new DoS attack where an adversary simply floods a user with bogus authentication attempts, effectively jamming their AIS transceiver from functioning in its primary navigational safety role. This could also potentially starve an entire geographic area of available transmission time on the network, limiting overall AIS transmissions.

In [30], an IBE scheme was proposed that follows IEEE 1363.3-2013 for Identity-Based Cryptography. Similar to [28], the authors propose a PKI system but use the vessel's MMSI number as the public key along with private keys obtained from a CA. They proposed several different modes to provide varying levels of security including anonymous authentication, public authentication, and symmetric key encryption. Their proposed authentication and encryption modes increase message overhead requirements are between 331 and over 700 bits. The most realistic advantage of this system is that vessels can simply derive an unknown vessel's public key from their MMSI number without consulting a database. However, there are significant issues with the large overhead they are including in their messages.

In [31], the design description of the USCG's Encrypted AIS (EAIS) system is discussed, which is an actual implementation and utilization of a government blue-force tracking tool that is currently in service. Similar to the proposals in [28, 30], this method uses AIS Messages 6 and 8 (binary messages) to transmit an AIS packet formatted similar to those set forth in the ITU standard. The specifics of each type of message are laid out in the document but remains largely unchanged from those shown in Chapter 2. The system includes three modes of operation: normal, receive-only and restricted. With all three modes both un-encrypted and encrypted transmissions are always received, meaning these modes only affect the type of outgoing AIS messages. Normal mode operates as a traditional AIS transceiver where all messages are sent out un-encrypted. In receive-only mode, the AIS transmitter maintains radio silence and does not transmit any messages. Finally, restricted mode encrypts all outgoing AIS messages. Transmissions remain on the normal AIS frequencies but restricted & receive-only modes do not permit the transceiver to be commanded to change frequencies. The format of message type 6 remains largely unaltered from the ITU standard, but the payload is encrypted using the Advanced Encryption Standard (AES). AES was adopted by the National Institute of Standards and Technology (NIST) as the U.S. standard symmetric block cipher. AES is extremely safe, and there are currently no known methods to break it. AES is capable of encrypting data in blocks of 128 bits [32]. The

EAIS system has the option to encrypt data using AES using 128-896 bits, which means packets are a total size of 144-912 bits. Depending on the length of the ASCII characters included in the data fields, this means the message will occupy between 2-5 TDMA slots. This system clearly works and meets the basic tenants of the security standards as it uses a government approved symmetric key encryption. However, symmetric key cryptography requires that all parties adhere to one standard, use one shared (and secret) key, and encrypt their entire messages. This would not scale to the public shipping community as the integrity of the symmetric key would come into question, as well as the ability to distribute and regulate such a system. Additionally, encrypted and confidential information is not a requirement nor a design feature of the AIS system. Government vessels require EAIS in order to conceal their position when operating in a law-enforcement capacity, but they maintain an increased burden and liability to monitor vessel traffic around them. These capabilities are not necessary for public shipping and actually defeat the original purpose of AIS, which was to be used as an enhanced situation awareness and collision avoidance tool. While this system is fairly straightforward to implement, it is not the right route to pursue for general AIS security.

In [5], Luft et al. from the U.S. Coast Guard Research and Development Center looked at methods to improve the performance of the AIS radio-link between fixed locations. When AIS was originally created, safety of navigation dictated that receiving corrupted data (invalid position/course/speed) was worse than receiving no data, so any message that failed CRC should be immediately discarded. In their research, they analyzed AIS from a surveillance perspective and developed a method to retain weak messages that failed the CRC using several physical-layer properties including time, frequency, antenna polarization, and radio path. These messages may be weak due to propagation or path-loss from distant targets. They identified that bit-stuffing can be problematic in these weak messages and opted to transmit a Message Type 26 without bit-stuffing and modified the receiver to match. Their system essentially acts as a repeater which packages AIS messages as the payload of a Message Type 26 and extends the range of AIS beyond the traditional VHF range without using satellites or other over-the-horizon (OTH) communications capabilities. This research provides several physical-layer avenues of research to explore which could provide further improvements in AIS security.

In [33], Kessler presents Protected AIS (pAIS) as a proof-of-concept to demonstrate that a PKI system similar to Mode 2 from [30] can be implemented within the current ITU technical standard, essentially allowing for immediate roll out to the existing maritime community. This system uses the private key of a sender to encode an 8-bit checksum over an entire AIS message and a message time stamp to generate a "protect string". As this message is now longer than the current expected AIS message, existing receivers would simply ignore the protect string extension while updated ones with pAIS would correctly decode the private key using the sender's public key and establish the authenticity of the message. While this implementation proves that backward comparability is relatively straight forward with this method, there are still several limitations that are not addressed. There is no geographic validity checking, which still means that a transmitter is capable of sending out GPS coordinates that could be false (assuming they have access to a valid PKI token). Additionally, since this was aimed as a proof-of-concept idea, the type and distribution of public/private keys is addressed only generally, and future work would require significant technical research into how this would be conducted, as it contributes significantly to the message overhead and thus the channel bandwidth.

3.2 NON-MARITIME INDUSTRY SECURITY

Due to the limited research into AIS, I turned to the aviation and vehicular network industries as they are currently facing similar challenges. The aviation industry's Automatic Dependent Surveillance-Broadcast (ADS-B) system suffers from many of the same vulnerabilities as AIS and due to the increased danger in their industry as well as a 2020 requirement to be carried aboard all commercial and military aircraft, there has been much more research conducted into mitigating these threats. In [34] ideas such as spread spectrum and frequency hopping, symmetric key cryptography, and PKI schemes are all analyzed as viable options but the authors conclude that they are difficult to scale due to issues with message overhead size and distribution and integrity of a key management system. Secure location verification is also identified as a way to correlate the position sent from an ADS-B transceiver by solving a geometric equation using time difference of arrival (TDOA) from several antennas located at known positions. While this solution would be feasible in the maritime domain since the geometric equation does not need to consider height as it would in aviation, this solution also lacks the ability to be scaled beyond internal and coastal waters, is subject to multipath propagation, and requires independent agency verification and notification of positional inaccuracies. Additionally, identifying the location of ships near the shore would be difficult due to the limited dimension of sensor location unless off-shore antennas were installed. The authors also proposed a retroactive key publication system based on the Timed Efficient Stream Loss-Tolerant Authentication (TESLA) protocol. TESLA is a scalable and loss-tolerant broadcast authentication protocol that uses message authentication codes (MAC) to validate the authenticity of uses [35, 36]. Unlike typical PKI systems, TESLA uses the time-delayed release of the key values used to generate a hash chain in a manner than mathematically ensures that all messages must have originated from the same source. Additional work has been done by the original authors to create a lightweight version called $\mu TESLA$ in [37] and to provide immediate authentication of packets instead of delayed buffering [38]. This has been applied for sensor networks such as in [39], vehicular networks in [40], and a recent practical implementation has been developed using SDR for ADS-B in [41]. To our knowledge, there has been no effort to adapt the TESLA protocol to the AIS system.

In [22], Ronald Reisman of the NASA Ames Research Center provides the description of a blockchain-based PKI framework to mitigate the security risks identified in the ADS-B aviation system. The idea is based on IBM's Hyperledger Fabric which is an open source blockchain platform. The author's goal was to implement a system that allowed for simultaneous secure and non-secure communications within the same channel through the use of "chaincode", similar to smart contracts used by the most popular open-source blockchain technology, Ethereum. Hyperledger fabric provides Distributed Ledger Technology (DLT) which allows for a more permission-based network as opposed to the permission-less systems used in other public blockchain technologies. Another important difference is that Fabric users must be enrolled in the network through a Membership Service Provider (MSP) which provides greater flexibility in providing more computationally efficient consensus algorithms such as Crash Fault Tolerance, rather than the BFT used in other anonymous services. Fabric also provides the ability to use a PKI structure that allows for authentication and encryption of communications. The authors note that the "on-ramp" issue, adding users on a global scale, will be the biggest hurdle to overcome as previous attempts have suffered from confusing international conventions. In their architecture, they attempt to overcome this challenge by using a Root CA and geographically distributed intermediate CAs (Peer Nodes) to handle the enrollment of end users. Every Peer Node holds an identical copy of all transactions, but unlike traditional blockchain, individual users are not required to maintain copies. This allows for private networks to be established where data can flow unrestricted from aircraft to Air Traffic Control, as well as within groups of aircraft such as military or civilian airlines. A prototype was also introduced that allowed aircraft within three nautical miles of each other to exchange safety of flight information. Hyperledger Fabric represents a very interesting research opportunity for AIS and could be applied to AIS in a similar manner to ADS-B where vessels are grouped to exchange data (military/government, commercial, recreational) with various permissions included. However, this solution would require an entire overhaul of the AIS data link from the ground up and would see a massive on-ramp issue in getting users to shift from the current AIS structure to a new blockchain based solution. Instead, I believe the biggest benefit of blockchain technology for AIS would involve using the distributed ledger to ease the burden of public/private key exchange and allow for vessels throughout the world to easily conduct authentication without concern for compromised keys or complex international regulations.

Research into Mobile Ad-hoc NETworks (MANET) is extremely pertinent to AIS since they function in a very similar manner to the way maritime vessels interact on the ocean. That is, the nodes exchange data between each other in a self-organized manner without interaction from outside entities. As shown in Chapter 2, the decentralized nature of PGP has led several MANET researchers to use it as a backbone to institute a PKI structure without the need for a CA. In [42], a self-organized PKI system is presented that allows individual users to generate public/private key pairs and authenticate other users. Authentication is performed through a chain of public-key certificates where a user checks the validity of each subsequent public key signature via the previous public key until that chain ultimately results in the validity of the desired end-user's public key. The lack of a CA comes at a cost, and that is felt through a significant data exchange with a lot of message overhead. For example, for two users to conduct authentication they must merge their certificate repositories, which they both had to store locally. Obviously as the network size increases, the storage required at each local user will become significant and exchange of that data could become quite a bottleneck. To overcome these large message overhead and bandwidth challenges, the authors in [43] present a PGP-like trust establishment scheme which uses certificate-less self-certifying IBE for authentication. Instead of exchanging trust-chains, this method allows individual users to compute another user's public key on demand, implicitly achieving self-certification of authenticity when a trust path exists. To use this model, if a node, n_i, trusts another node, n_j, n_i issues a "witness" W_{ij} over a secure channel. n_j uses W_{ij} to generate its private key, cryptographically binding W_{ij} to the identity of n_j. Since this is an IBE scheme, any node that trusts n_i can compute the public key of n_j. There are two levels of trust assumed here, level 1 and level 2, whose indices generate a "trust graph". Level 1 trust means that a node trusts the public key of another node, while Level 2 trust means a node trusts another to issue witnesses and recommend other nodes. Trust values here are in the range $[0, 1]$ and represent the reliability of a node where higher values equate to more trust. As users in the network interact, weighted trust paths are established, reducing the ability of malicious intermediate nodes to completely derail the system by falsifying public keys. The authors here used ECC for key generation and found that compared to [42], their model was much more computationally efficient and required significantly less overhead. For example, with 500 users and a path length between users of 20, certificate-less web-of-trust required .25 KB storage, compared to 1500 KB in [42]. Additionally, total communication cost was reduced from 733MB in [42] to 140 KB in certificate-less web-of-trust. While traditional PGP-style schemes would be unable to scale to the millions of vessels using AIS every day, this scheme uses both IBE as well as a decentralized infrastructure and may be computationally efficient enough to use within the AIS architecture. However, this would likely require significant changes to the AIS technical standards and may not be backward compatible with existing transceivers. This research may be worth looking into in the future to deal with maritime public/private key exchanges and eliminate the need to store a multitude of public keys.

CHAPTER 4

SOFTWARE DEFINED RADIO IMPLEMENTATION OF AIS

In this section, I will provide a brief background on Software Defined Radio and then explain how I used this tool to create a working AIS transmitter and receiver that provides a highly robust test platform for researching improvements in the AIS system.

4.1 SOFTWARE DEFINED RADIO BACKGROUND

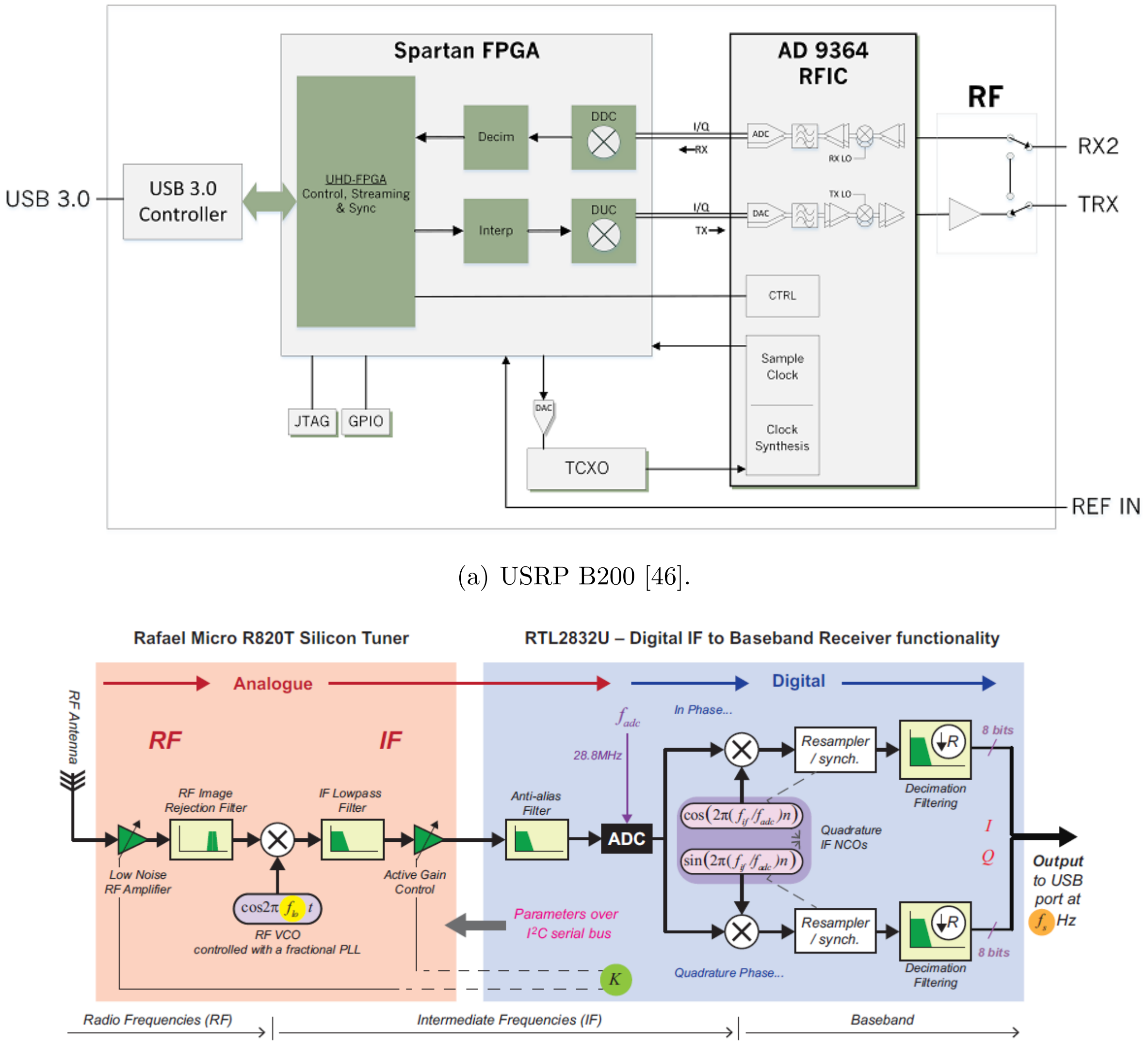
In the early 1990s, Joseph Mitola first described the architecture details of software radio, or Software Defined Radio (SDR), as a device that can easily reconfigure itself to select the best transmission mode and adapt to the current environment based on cost, service availability, or signal quality [44]. SDR allows for physical layer hardware components of a radio system to be implemented using software, which allows for a highly customizable and reconfigurable system that can be changed "on the fly". Ideally, the only hardware components that would need to be included in a SDR are the antenna and a high-speed sampler [45]. For my research, I used a USRP B200 as my transmitter and a RTL-2832U as my receiver. I also used the GNU Radio Software to build the radio block diagrams which simulate the radio hardware components.

4.1.1 USRP B200

The Universal Software Radio Peripheral (USRP) B200mini by Ettus Research is a SDR that retails for approximately $902.00 USD. This device comes with USB 3.0 SuperSpeed connectivity and can be interfaced using USRP Hardware Driver (UHD) open-source software version 3.9.0 or later. There are three inputs which can be seen in the block diagram of the device in Fig. 15(a): transmit, receive, and reference [46].

The features of the B200 are as follows:

- Frequency range: 70 MHz-6 GHz
- Full duplex operation with 56 MHz instantaneous bandwidth (61.44 MS/s quadrature)
- Open and reconfigurable Spartan 6 XC6SLX75 FPGA with free Xilinx tools
- Gain: 76dB available at receiver and 89.8 dB available at transmitter



(a) USRP B200 [46].

(b) Nooelec NESDR Smart RTL [45].

Fig. 15: Internal block diagram comparison of USRP and RTL SDR.

4.1.2 NOOELEC NESDR SMART RTL-2832U

The NooElec NESDR RTL is a much more inexpensive SDR aimed at the amateur radio market and retails for only $29.95 USD. This device is approximately the size of a stick of gum and is only capable of handling one antenna in a receive-only mode with a much smaller bandwidth than the USRP. An internal diagram of the RTL device can be seen in Fig. 15(b).

The features of the device are as follows [47]:

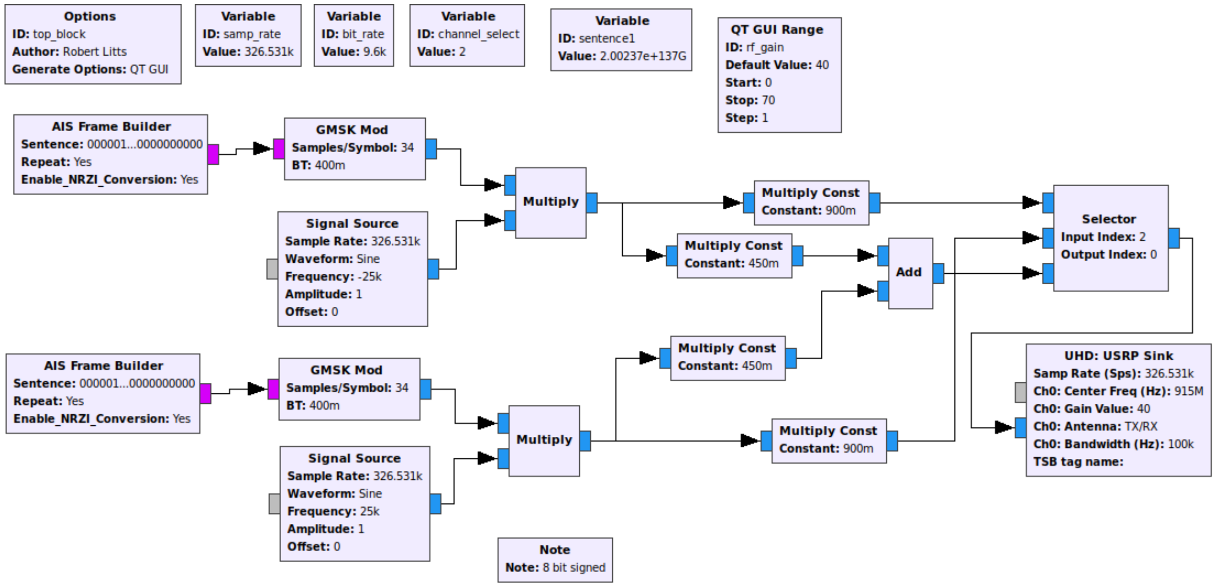
- Frequency range: 25MGz-1.75GHz
- Aluminium enclosure
- SMA female antenna input
- 2.4MHz (nominal) and up to 3.2MHz (max) bandwidth
- Gain: 29 settings from 0-49.6

4.1.3 GNU RADIO COMPANION

GNU Radio Companion software is a powerful graphical user interface tool that allows for a robust implementation of signal processing tools. GNU Radio began as a project at MIT in 2004 with Matt Ettus as one of the first developers who created the USRP hardware platform for use with GNU radio software. In 2009, Josh Blum distributed the GNU Radio Companion (GRC) software at the annual “Hackfest”. It was a “drag and drop” software front-end that acts as a Python code-generator when compiled [48]. I used GNU Radio 3.7.13.5 on a PC running the Linux Ubuntu operating system.

OpenCPN Chart Plotter

Since this research was conducted about a maritime collision avoidance device, I naturally needed a method to display the data I received in a similar manner to the electronic chart display information system (ECDIS) software used about most commercial vessels. To do this, I used OpenCPN which is an open source chart plotter that contains significant documentation on their GITHub page [49]. OpenCPN allows for the installation of current nautical charts, although these are not necessary to see the AIS data. Instead, they provide a more realistic view of vessel locations especially when plotting real ship traffic.



Options
ID: top_block
Author: Robert Litts
Generate Options: QT GUI

Variable
ID: samp_rate
Value: 326.531k

Variable
ID: bit_rate
Value: 9.6k

Variable
ID: channel_select
Value: 2

Variable
ID: sentence1
Value: 2.00237e+137G

QT GUI Range
ID: rf_gain
Default Value: 40
Start: 0
Stop: 70
Step: 1

AIS Frame Builder
Sentence: 000001...0000000000
Repeat: Yes
Enable_NRZI_Conversion: Yes

GMSK Mod
Samples/Symbol: 34
BT: 400m

Signal Source
Sample Rate: 326.531k
Waveform: Sine
Frequency: -25k
Amplitude: 1
Offset: 0

Multiply

Multiply Const
Constant: 900m

Multiply Const
Constant: 450m

Add

Selector
Input Index: 2
Output Index: 0

AIS Frame Builder
Sentence: 000001...0000000000
Repeat: Yes
Enable_NRZI_Conversion: Yes

GMSK Mod
Samples/Symbol: 34
BT: 400m

Signal Source
Sample Rate: 326.531k
Waveform: Sine
Frequency: 25k
Amplitude: 1
Offset: 0

Multiply

Multiply Const
Constant: 450m

Multiply Const
Constant: 900m

UHD: USRP Sink
Samp Rate (Sps): 326.531k
Ch0: Center Freq (Hz): 915M
Ch0: Gain Value: 40
Ch0: Antenna: TX/RX
Ch0: Bandwidth (Hz): 100k
TSB tag name:

Note
Note: 8 bit signed

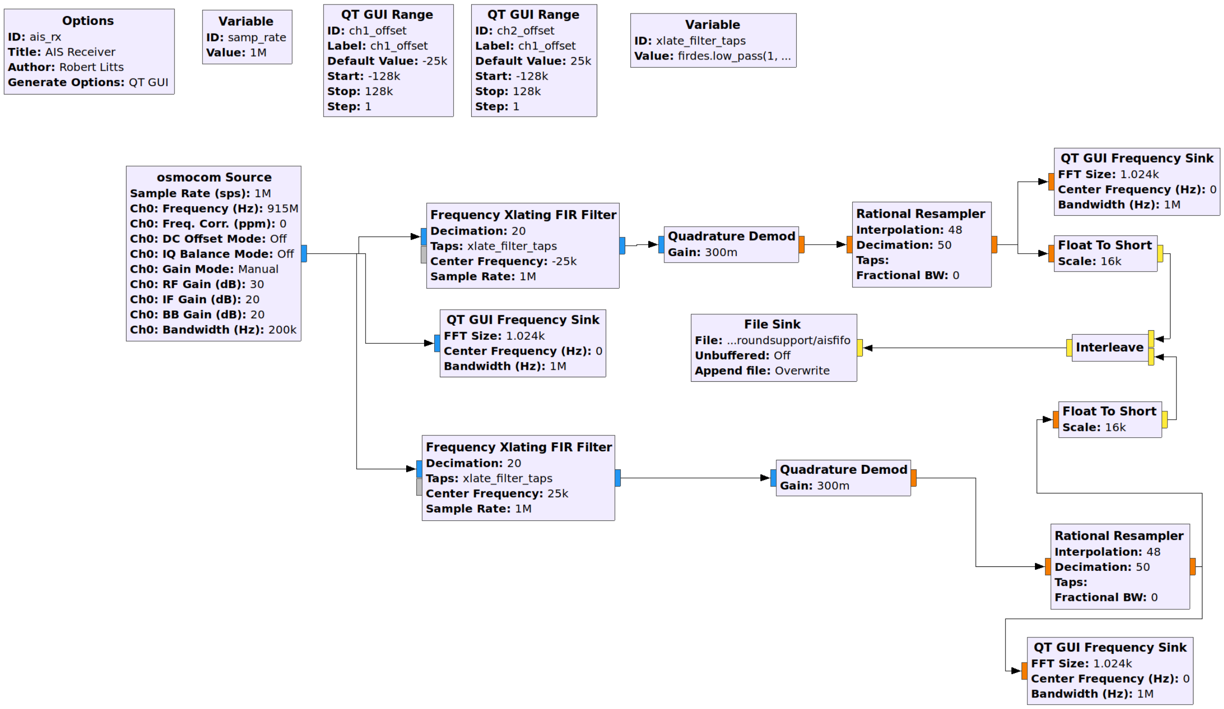
Fig. 16: AIS Transmitter Block Diagram using GNU Radio

4.2 SDR AIS TRANSMITTER AND RECEIVER

Using GNU Radio, I built upon the work of [25] to design and test a wireless AIS transmitter and receiver that is capable of sending and receiving real AIS packets, decoding them, and plotting them on an ECDIS style chart plotter using OpenCPN. First, the AIS transmitter remains largely unchanged from [25], but the authors did not connect their setup over-the-air and instead opted to use a wired connection for transmission. The designed transmitter is shown in Fig. 16

I opted to conduct transmissions in the unlicensed 900MHz frequency band to avoid any potential interference with actual AIS transmissions. However, I did test my AIS receiver using the default AIS channels 1/2 of 162MHz at a nearby river at the Great Bridge Lock in Chesapeake, VA and was able to receive recreational and commercial AIS transmissions directly to my laptop. Note that the “sentence” within the AIS Frame Builder block is a binary string of data converted from an AIVDM message containing the correct parameters for the desired message type to be sent. At the receiver, the Osmocom NESDR Smart RTL-SDR is tuned to 915MHz to receive the signal. This is shown in Fig. 17.

The signal is then sent to two separate FIR filters at Channel A and B, followed by a quadrature demodulation, and then down-sampling to 48,000 samples/second. Finally, this data is interleaved and sent to a file sink which contains a named UNIX pipe. This UNIX



Options
ID: ais_rx
Title: AIS Receiver
Author: Robert Litts
Generate Options: QT GUI

Variable
ID: samp_rate
Value: 1M

QT GUI Range
ID: ch1_offset
Label: ch1_offset
Default Value: -25k
Start: -128k
Stop: 128k
Step: 1

QT GUI Range
ID: ch2_offset
Label: ch1_offset
Default Value: 25k
Start: -128k
Stop: 128k
Step: 1

Variable
ID: xlate_filter_taps
Value: firdes.low_pass(1, ...

osmocom Source
Sample Rate (sps): 1M
Ch0: Frequency (Hz): 915M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Off
Ch0: IQ Balance Mode: Off
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 30
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20
Ch0: Bandwidth (Hz): 200k

Frequency Xlating FIR Filter
Decimation: 20
Taps: xlate_filter_taps
Center Frequency: -25k
Sample Rate: 1M

Quadrature Demod
Gain: 300m

Rational Resampler
Interpolation: 48
Decimation: 50
Taps:
Fractional BW: 0

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 1M

Float To Short
Scale: 16k

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 1M

File Sink
File: ...roundsupport/aisfifo
Unbuffered: Off
Append file: Overwrite

Interleave

Float To Short
Scale: 16k

Frequency Xlating FIR Filter
Decimation: 20
Taps: xlate_filter_taps
Center Frequency: 25k
Sample Rate: 1M

Quadrature Demod
Gain: 300m

Rational Resampler
Interpolation: 48
Decimation: 50
Taps:
Fractional BW: 0

QT GUI Frequency Sink
FFT Size: 1.024k
Center Frequency (Hz): 0
Bandwidth (Hz): 1M

Fig. 17: AIS Receiver Block Diagram using GNU Radio

pipe is used to send the data from GNU Radio to GNU AIS, which is an open source AIS program that converts the received AIS message into a format readable by the chart plotter. To correctly receive the data, the output file must be a unix pipe which can be created using the following command:

mkfifo aisfifo

This generates a unix pipe called "aisfifo" and the output file in the GRC file sink should be pointed to the location of where this pipe was created. Now the GRC flowgraph can be run with the data sent to the named unix pipe, followed by the GNU AIS program from the command line which will route the data to OpenCPN chart plotter. This can be done by following the command below:

gnuais -l aisfifo -c $\tilde{/}$*.config/gnuais/config*

The $-l$ command provides the program with the name of the source file to read from, while the $-c$ command provides it with the location of the configuration file where you can configure the data to be sent to another location. I modified this configuration file to send the data through a virtual serial port so it can be displayed on OpenCPN. To create a virtual serial port, I used the *socat* program on Linux using the following commands:

```
socat -d -d pipe:ais _ pipe pty &
```

This generates a named pipe called `ais_ pipe`. The output of this command will tell you the name of the virtual serial port, which should be of the form: /dev/pts/6. To send the AIS data to OpenCPN, you must configure the virtual serial port on OpenCPN. Within OpenCPN, you must go to 'Options', 'Connections', and then 'Add Connection' and then choose a Serial connection and input the name of the Serial port from the previous step which was obtained using the 'Socat' command. Using the above example, the serial port would be input as /dev/pts/6.

The final step is to edit the configuration file that gnuais uses so that the received data can be sent to the virtual serial port. Note: there are two pipes being used in this scenario. The first pipe, called aisfifo sends data from GNU Radio to GNU AIS. The second pipe, called `ais_ pipe` (created using socat) sends data from GNU AIS to OpenCPN. To edit the configuration file, complete the following: Navigate to the config file for GNU AIS and there should be a default line commented out that says *# serial _ port /dev/ttyS0*. Uncomment this and change it to match the serial port you created using the socat command. In my example above, this would read as follows:

```
serial_ port /dev/pts/6
```

Additional configuration can be done in this file such as sending the data to a SQL database or server for further processing. Now that this is installed and serial ports are configured, running the transmitter and receiver in unison should see your vessels populating on the chart. The latitude and longitude I chose are specific to the Chesapeake, VA area.

Based on these tests, it is clear that there is absolutely nothing that prevents anyone with a SDR from impersonating any vessel on the ocean and completely falsifying their position, course, and speed. Additionally, using this same methodology you can generate fake ATON and send Channel Management and Group Assignment commands which would effectively cut users out of the AIS data link altogether. The simple fact that there is nothing stopping an individual from transmitting fake AIS targets is alarming and validates vulnerabilities 1-9 from Chapter 2. Without proper authority, it would be unethical to showcase these vulnerabilities using actual vessels, but the possibility exists that an individual armed with

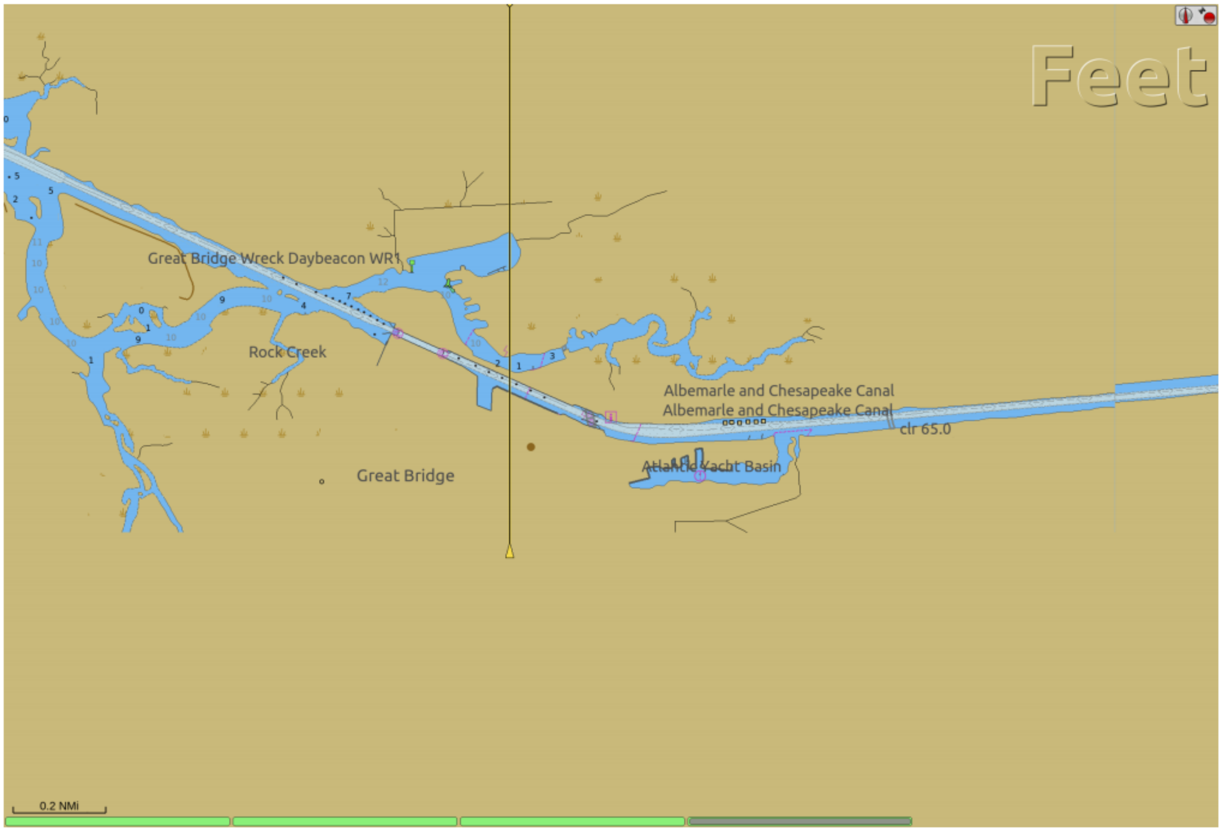

Feet
Great Bridge Wreck Daybeacon WR1
Rock Creek
Albemarle and Chesapeake Canal
Albemarle and Chesapeake Canal
clr 65.0
Atlantic Yacht Basin
Great Bridge
0.2 NMi

Fig. 18: Spoofed AIS Target Transmitted from SDR Transmitter to SDR Receiver and plotted in OpenCPN

a SDR and a VHF antenna in a major harbor or congested waterway could cause significant disruptions or even a collision. The simple idea of confusing a VTS in a busy waterway such as New York Harbor could potentially be enough to cause individuals to lose concentration and allow a collision to occur. Based on the work here, I believe it is imperative that user authentication and integrity be included in the AIS message protocol. By including these two crucial security features, there would be no way for me to transmit data as any MMSI, since a receiver could simply receive a notification that the sender is not authentic. Data encryption is not a necessary feature since AIS data should remain public; every vessel on the ocean should be able to know the position, course, and speed of all others around them; there is no secret that must be shared. However, every mariner deserves to know that the data received from the vessels around them is authentic, so a solution must be implemented which can seamlessly include this feature without obscuring the use of AIS as a publicly available collision avoidance tool.

CHAPTER 5

AIS WITH AUTHENTICATION

Previous works to update the security of AIS have all focused on an entirely PKI based solution that necessarily includes significant increases in packet sizes and subsequently the number of consecutive message slots required for an individual to send autonomous position reports. AIS operates in a bandwidth constrained environment with only 2,250 time slots available every minute (4,500 using both channels), so keeping packet sizes to a minimum should be a key goal of any AIS authentication protocol. Therefore, I have looked to aviation and other transportation industries in order to shed light on alternative methods that can be adapted to the unique challenges of the maritime environment. The Timed Efficient Stream Loss-Tolerant Authentication (TESLA) protocol was referenced as a possible solution for ADS-B security and functions as an asymmetric cryptography system without the need to protect and share secret keys between users, reducing message exchange and ultimately freeing up TDMA slots for other transmissions. In this section, I will discuss an update to the AIS system that embeds a HMAC onto AIS packets that were generated from a pseudo-random function (PRF) using keys initially known only to the sender and are periodically broadcast to all stations. Since the authentication protocol implied by TESLA involves only software processing of the data bits, it can be implemented by adding a software update to the existing AIS software to process the authentication packets. Specifically, the existing AIS software would handle physical layer data conversion (GMSK demodulation and NRZI decoding) followed by processing of the additional link layer functions if the new type of authentication packets is detected, as shown in Fig. 19. If no authentication packets are detected, which is an indication that a user has not yet updated the AIS to include authentication, the new authentication component of the updated AIS system is bypassed and the AIS software would function as it currently does, with an additional note on the vessel's chart plotter/RADAR mentioning that “authentication is not available”.

In the following section, I will describe the premise of the TESLA protocol which will allow for better understanding of how it can be adapted for use in the AIS system.

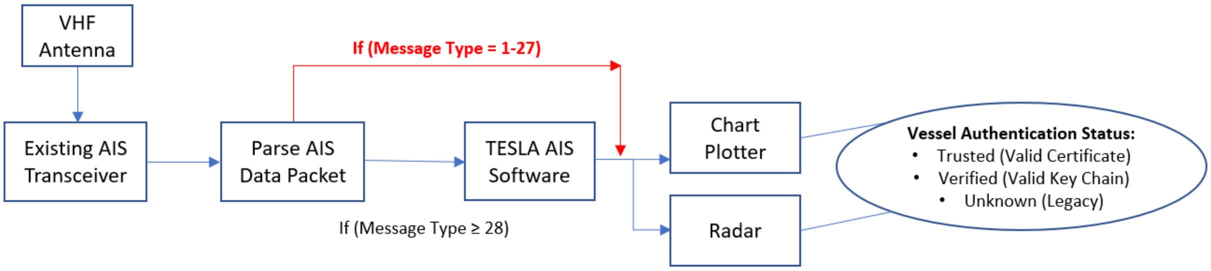


Fig. 19: Interfacing the TESLA-based authentication component with the existing AIS.

5.1 TIMED EFFICIENT STREAM LOSS-TOLERANT AUTHENTICATION PROTOCOL

The TESLA protocol was introduced in the context of multicast communications to enable receivers to verify that the received data originated with the claimed source and was not modified as it transited through the network [35, 38, 36]. To accomplish this task, TESLA replicates asymmetric cryptography principles such that a receiver can authenticate the source of a message without being able to reproduce the authenticated message. The requirements of the TESLA protocol include:

- Loose time synchronization of users, which is satisfied by the universal time coordination (UTC) feature of the existing AIS system.
- Access to a pseudo-random function (PRF) family or "one-way function", F such that $F(k) = x$, such that given x, k cannot be back-computed and F cannot be distinguished. Additionally, given k, $F(k)$ will always produce the output x.

To begin the protocol, the sender will choose a random value, K_n to begin a PRF chain of length n. Using PRF F, the sender first computes $K_{n-1} = F(K_n)$ which is the initial commitment to the PRF chain. The remaining keys up to K_0 are computed using this same format, essentially computing F of every value in a chain. This proves one of the most important principles of TESLA: loss tolerance. Since $K_{n-1} = F(K_n)$ and each K is produced recursively using F, then any receiver who receives any K_n value is able to produce all prior key commitments. However, the properties of the PRF mean that this same receiver will be unable to forward-compute any other keys and thus would be unable to replicate the sender. The sender will also determine a time release schedule, Δt, for which each of the K_n will be released, with each key being active during one time interval. Starting at time T_0, the sender will release the first key K_0 and then release K_1 at $T_1 = T_0 + \Delta t$, and so on

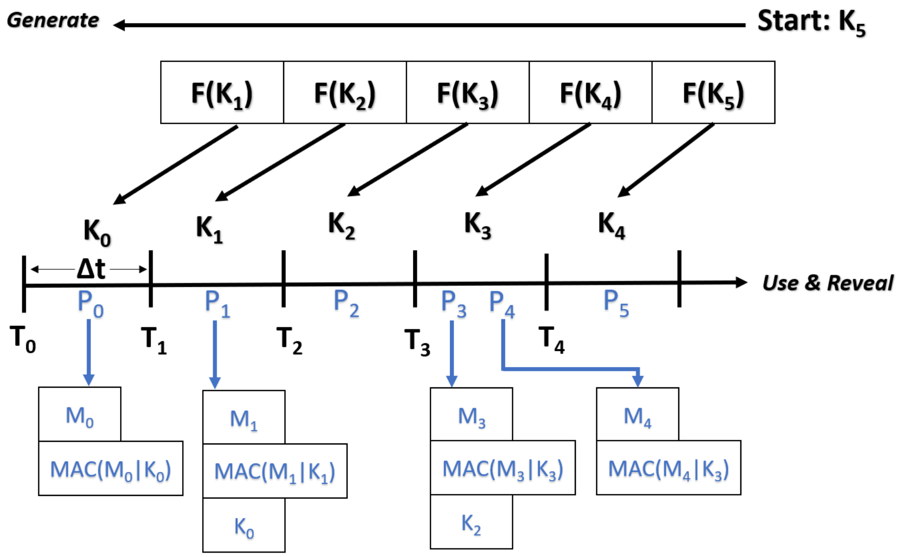


Fig. 20: TESLA Broadcast Authentication Protocol: Key Generation, Use, and Reveal, $n = 5$.

until all n keys are released. If any receiver misses the release of a key during a certain time interval (due to packet loss or other conditions), they can simply derive prior keys using the PRF to validate prior messages. It is important to note that while this system establishes a sound link between the initial message released and all messages in the chain, this only proves that messages all originated from the same source. In order to prove the identity of that source, some form of PKI validation must occur. We will assume that the source of the first key has been validated as authentic, and thus all remaining keys derived from this must also be valid. The TESLA protocol can be seen in Fig. 20.

The sender must broadcast their Δt and T_0, as well as the first key in the key chain commitment, K_0. This is done in time interval T_i by sending a packet, P_j such that $P_j = [M_j|MAC(M_j|K_i)|K_{i-1}]$. This means for the current time interval, the sender is broadcasting a plain text message along with the previous time interval's key and has appended a MAC comprised of the next time interval's key and the message itself. This will allow the user (upon receipt of the subsequent packet) to verify the integrity of the P_j. Since this is a broadcast protocol, all receivers will obtain P_j, which means that they have to ensure that only the sender could have produced the message using information available to them prior to the disclosure interval. Using the initial time delay schedule published at the start of their broadcast transmissions, the receiver can calculate the current time interval and verify that the message they received was not sent after $T_0 + i * \Delta t$. If it was, anyone

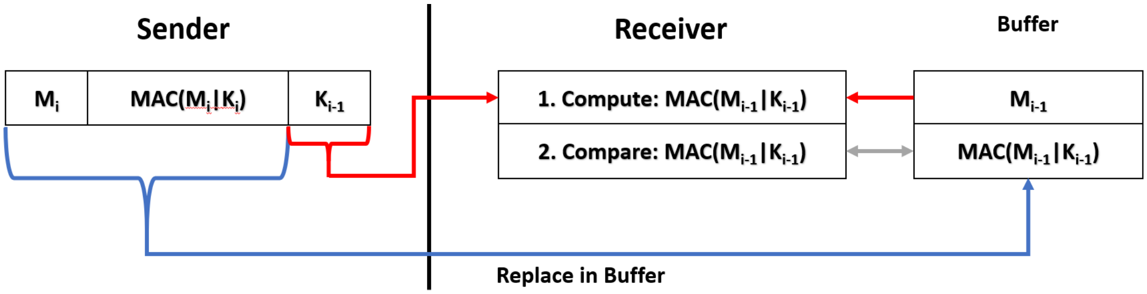


Fig. 21: TESLA Sender and Receiver Overview.

could have sent that message (since the key is now public) and thus the receiver would discard the message as unauthentic. If the time interval is valid, then P_j can be validated using the key from the previous interval. If this is the first message from a specific user, then the receiver simply stores this info to a buffer and awaits for the next transmission. If the receiver already has a key in the buffer, then they can use the key they just received to compute the $MAC[K_{i-1}|M_{i-1}]$ and it should be equivalent to the MAC in the prior message that was stored in the buffer. This process can be seen in Fig. 21.

5.2 SYSTEM MODEL AND PROBLEM STATEMENT

In this section, I consider an update to the AIS system which embeds a TESLA MAC onto AIS packets that were generated from a PRF chain initially known only to the sender with keys periodically broadcast to all stations. Since AIS is already a UTC time synchronized protocol, this would allow the creation of an asymmetric cryptography scheme where the time-delayed release of PRF keys are used to verify the source and integrity of prior messages. This design would implement new message types (beyond the current 27 currently in use) that contain a minor modification to the existing data sent within an AIS frame. This would essentially use Message Type 28-Message Type 55 to create authenticated versions of existing Message Types 1-27. For example: Message Type 28 would be an authenticated version of Message Type 1 (scheduled position report), Message Type 29 would be an authenticated version of Message Type 21 (ATON report). This would also allow the system to be implemented as a software-only solution that can be adapted to current AIS transceivers and immediately interact with the maritime community. Additionally, modern hash algorithms meet the requirements of a PRF and will not remain “secret” for very long periods of time, so the likelihood of conducting even a brute-force attack on any one chain would be almost impossible for the 80-bit hash function proposed. Although this paper does not attempt to delve into a comparison between various hash function families and their associated mathematical proofs, it is sufficient to say that modern hash functions, even truncated in length, provide more than adequate security over the extremely short time spans that have been used in this paper.

The model requires the following key participating users:

- Central Authority (FCC, IMO, or other national licensing agency to provide and distribute private keys)
- Vessel A, Transmitting Ship
- Vessel B, Receiving Ship, within VHF range of Vessel A

The benefits of TESLA AIS system include:

- Resistant to packet loss
- Limited overhead size: Rapid changing of cryptographic hash functions means less data needs to be used per hash
- Minor updates to current packet structure
- Can be implemented as a software-only upgrade to existing AIS systems using spare, unused message types defined by ITU (i.e. legacy transceivers can simply check the message type and simply pass it through if it is not applicable)

This model contains the following modes of operation, which will be further explained in detail:

- Setup: Generation of hash chain and key delay schedule
- Online: Broadcast delayed disclosure of hash chain keys over VHF channel
- Receiver Verification: Validate the identity of users
    - Confirm message type (to confirm compatibility or pass legacy messages straight through to original system)
    - Validate digital signature of K_0 to confirm identity, and if so, user is **Trusted**
    - Compute hash of current key and compare to previous keys to verify trusted user has sent messages, user is **Verified**

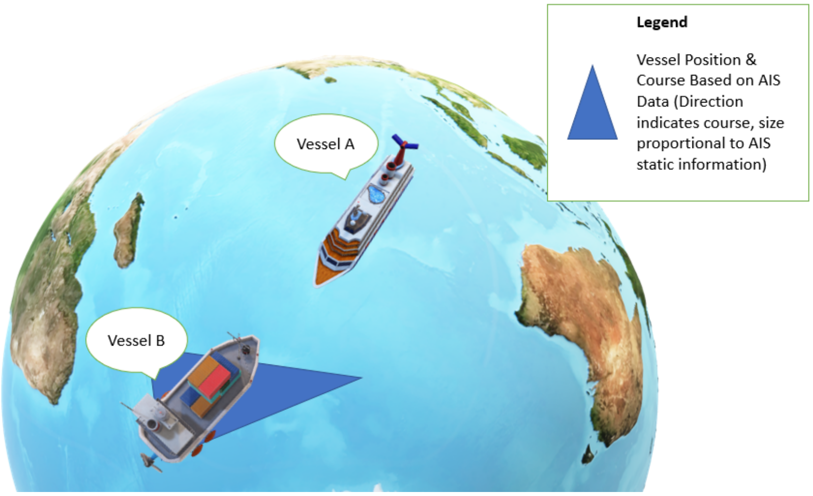

Legend
Vessel Position & Course Based on AIS Data (Direction indicates course, size proportional to AIS static information)
Vessel A
Vessel B

Fig. 22: Two vessel interaction using the existing AIS infrastructure; Vessel B is transmitting incorrect AIS data and Vessel A has no choice but to assume it is accurate.

The goal of our TESLA authentication scheme is to provide existing AIS users with a way to validate the source of incoming AIS messages and ensure they have been unaltered by an adversary while limiting individual message sizes to ensure multiple access to many simultaneous users. Using the existing AIS framework, a typical encounter on the ocean occurs as depicted in Fig. 22.

Vessel A is transiting international waters and comes within VHF range of Vessel B. Vessel A's country of origin is Panama, and it has never encountered Vessel B, whose AIS static data shows that they hail from Spain. Since the vessels are not in sight of one another, Vessel A cannot determine whether the data they are receiving from Vessel B is accurate and therefore must trust that they are accurately reporting their information. International standards require both vessels to ensure their transmissions are accurate, but enforcement of these standards is subject to the vessel's flag state which results in various levels of adherence to the rules. As Vessel A and Vessel B continue to transit toward one another, Vessel B's AIS data is used onboard Vessel A to compute the closest point of approach (CPA) which is a measurement used to determine maneuvers based on international navigation rules. As Vessel B comes within sight of Vessel A, the watch keeper notices that there are differences between what they see visually and what the AIS data shows (different vessel name, size, course). The watch keeper now devotes extra time to observing this vessel until they are past and clear due to their loss of confidence in the information they have received about the vessel.

In the updated authentication protocol, this same encounter would occur as follows: Vessel A is transiting international waters and comes within VHF range of Vessel B. Vessel B broadcasts a position report that contains a digitally signed hash chain key commitment and a MAC. Vessel A receives Vessel B's position report, retrieves Vessel B's public key from a database and upon receipt of the next message from Vessel B, compares the contents of their digital signature with the MAC from their previous message using the new key chain sent by vessel B in their latest message. On Vessel A, the public key retrieved from the database associated with the vessel's MMSI does not validate the private key Vessel B has used to signed their message, and the system returns an error message stating that "Vessel B is not trusted." This is due to the fact that Vessel B is not actually the vessel associated with the MMSI they are broadcasting; if they were, their position reports would be signed with the proper private key. Vessel A now knows that the identity of Vessel B cannot be confirmed, so they can proceed to steer clear of this vessel in order to avoid negative interactions.

As shown here, an authentication protocol greatly improves the confidence a vessel operator has in the data they are receiving and alerts the user when data should be subject to scrutiny. In the next section, I will discuss how this system can be implemented to operate within the existing AIS framework.

5.3 IMPLEMENTING TESLA IN AIS

As discussed in Section 5.1, implementation of the TESLA protocol requires senders to have a set message delay disclosure interval, Δt, for the release of subsequent keys. However, multiple messages can be sent in one time interval using the same key, meaning that multiple messages would need to be stored on the receiver's end at one time, but we do not expect this storage requirement to be significant. To calculate the storage, we must consider the reporting interval required of users. The maximum nominal reporting interval that could be required by any vessel is a Class A mobile user operating at fast speeds and changing course, and requires position updates every 2 s, or about once every 75 time slots [1].

The use of SOTDMA in the AIS facilitates including the TESLA protocol for authentication. When operating in autonomous and continuous mode, SOTDMA allows the $2,250$ available TDMA frames within one UTC minute to be assigned without conflict by using a slot offset counter to identify the number of frames remaining for transmission by the current user. Upon first entering the network, the AIS user will follow an initialization protocol that monitors the link for 1 minute to create a map of the transmission slots currently in use by other users. In addition, the following terms are used to address transmission within the link:

- The reporting interval, between 2 s and 30 s

  $$2 \leq RI \leq 30.$$

- The number of position reports required per minute

  $$RR = 60/RI.$$

- The number of slots before the user will need to transmit

  $$NI = 2250/RR.$$

- The first slot used to announce entry to link NSS.
- Slot number selected for position reports

  $$NS = NSS$$

  for first transmission in frame, and

  $$NS = NSS + (n * NI), 0 \leq n \leq RR$$

  for subsequent frames.

- The collection of possible slots for a position report

  $$SI = [NS - (0.1 * NI), NS + (0.1 * NI)].$$

At a minimum, a user transmitting once every 30 seconds would require $NI = 1125$, which implies that the user would need to transmit its position once every 1125 slots (once per UTC minute). With a randomly chosen nominal transmission slot NTS, from within the possible SI values, the distance between frames will not always be spaced the same number of slots apart. To overcome this challenge, we can allow $\Delta t = 1$ min for all users, which implies that all messages sent within the same UTC minute contain a MAC with the same key, and allows key disclosure to occur in conjunction with the UTC minute in order to synchronize key disclosures by all users. Aside from vessels moored or at anchor (not moving), all vessels operating in autonomous mode report their position at least once per minute. Therefore, using $\Delta t = 1$ min, each user would disclose the key to decode all messages from the previous UTC minute within their first transmission of the current UTC minute. This would simply mean that a receiver will need to buffer up to 30 messages at a time if a sender is operating at the maximum reporting rate of 2 s reporting. One potential problem with this approach is that up to 30 messages may have been sitting in a buffer and have not been viewed by the user, resulting in reporting delays in the AIS. As the AIS information is extremely time sensitive due to its use in collision avoidance, we note that the buffered data on the receiver side should be initially assumed to be accurate until proven otherwise and should be immediately used for course plotting as the system would still maintain the ability to retroactively flag unauthentic data from the prior minute. As an alternative, Δt may be reduced to RI and the sender required to initially publish their NSS upon admission to the AIS network. This would allow key disclosure to be conducted at each NTS making reports delayed by only the reporting rate. The receiver would need to verify that the received message was sent prior to the NTS plus a factor of the high bound of the SI. This would require minimal additional overhead from the sender but would allow for more rapid message authentication at a delay equal to the reporting rate. To ensure authentication at the same rate as initially intended by the ITU standard, reporting rates would need to be increased to twice their current rate. For example, a unit currently reporting once every 2 s would need to use $\Delta t = 1$ s in order to generate an authenticated position at the receiver every 2 s. It is also worth noting that the length N of the authentication hash chain is not a significant factor for implementing the TESLA authentication protocol in the AIS, and that an authentication hash chain of length N only requires $\log_2(N)$ storage and can be computed with that same amount of power. Depending on the key disclosure rate, more keys may be necessary, which is an important factor in determining N. If keys are disclosed at the maximum reporting rate of once every 2 s, this would require 30 keys per minute or 1800 keys per hour. Using 10 byte keys, this would imply an additional storage requirement between 30 bytes and 18 KB of data to store. Therefore, storage is not a significant concern since hash chains are fixed length and do not grow exponentially in size as the chain grows larger. In order to provide the most robust and up-to-date data for the AIS, $\Delta t = RI$ and the chain length, $N = (RR * 60) + 6$, where $126 \leq N \leq 1,806$, meaning that key chains are recomputed at most once per hour. Since the RI is a function of a vessel's navigation status, speed, and course rate-of-change and static information is sent every 6 minutes (see Ref. [1]), Δt can be implied by the receiver without the need to formally exchange this data. In addition, since AIS messages do not include a time stamp (aside from UTC seconds in

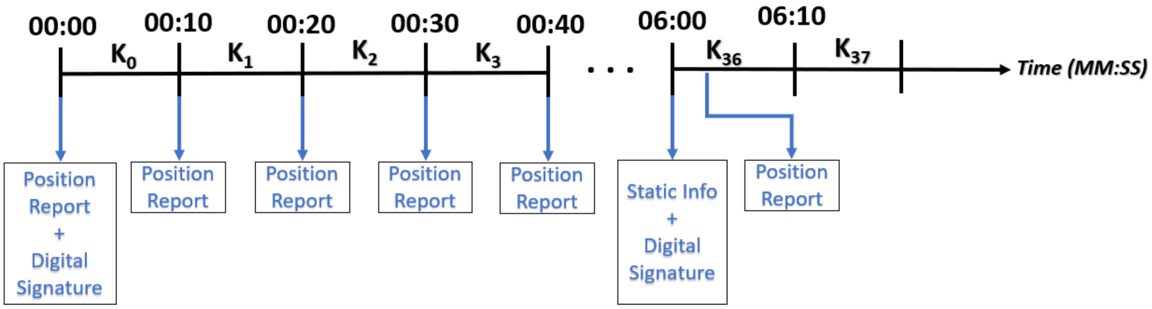


Fig. 23: AIS with Authentication using TESLA keys embedded in existing position reports.

Type 1-3), this must be included in order for the receiver to determine if a message has arrived within the correct Δt. The current date is also included as part of the MAC to create a unique message and prevent replay attacks. Another important point is that when the receiver validates the MAC, they are also confirming the integrity of the message.

We also use PKI digital signatures to provide authentication of one element of the key chain which allows the receiver to confirm the identity of the sender. To do this, we use the sender's private key (signed by a CA and known only to them), $S_{-}(\cdot)$, to digitally sign the initial MAC containing K_0 in M_0. Upon receipt of this message and the subsequent key disclosure of K_0 in M_1, the receiver can use the sender's public key, $S_{+}(\cdot)$, to authenticate the sender. Additionally, since all previously sent keys can be computed from any one key, a receiver who misses this initial broadcast containing the digital signature can simply request a digitally signed key chain commitment from the sender and could authenticate all past messages they have received from the sender, as long as all MACs were validated and arrived within the proper time constraints.

Finally, although this protocol only addresses automatic position reports, the concept can easily be expanded to include additional vessel types. Static AIS data (Message Type 5) which is transmitted every 6 minutes, could easily be included in this framework by embedding a MAC onto the message and subsequently disclosing the key in a follow-on position report, thus linking that report to the hash chain. Message Type 5 can also be used to provide authentication for those vessels that were not in range for a sender's initial digitally signed broadcast position report. This means that Message Type 5 would provide a periodic broadcast of a vessel's digital signature and would allow frequent authentication by new users who enter the link. A representation of how this would work within the AIS architecture is shown in Fig. 23.

5.4 SIMULATIONS

The proposed authentication protocol for the AIS was tested using Python 3 as a software front-end, which was paired with GNU Radio for controlling SDR implementations of the AIS transmitter and receiver. The pseudocode for the Python front-end includes the start up information for a user, the broadcast message creation, and the actions taken by the receiver to validate the contents of a message.

Algorithm 1 – AIS Transmitter: Authentication Startup

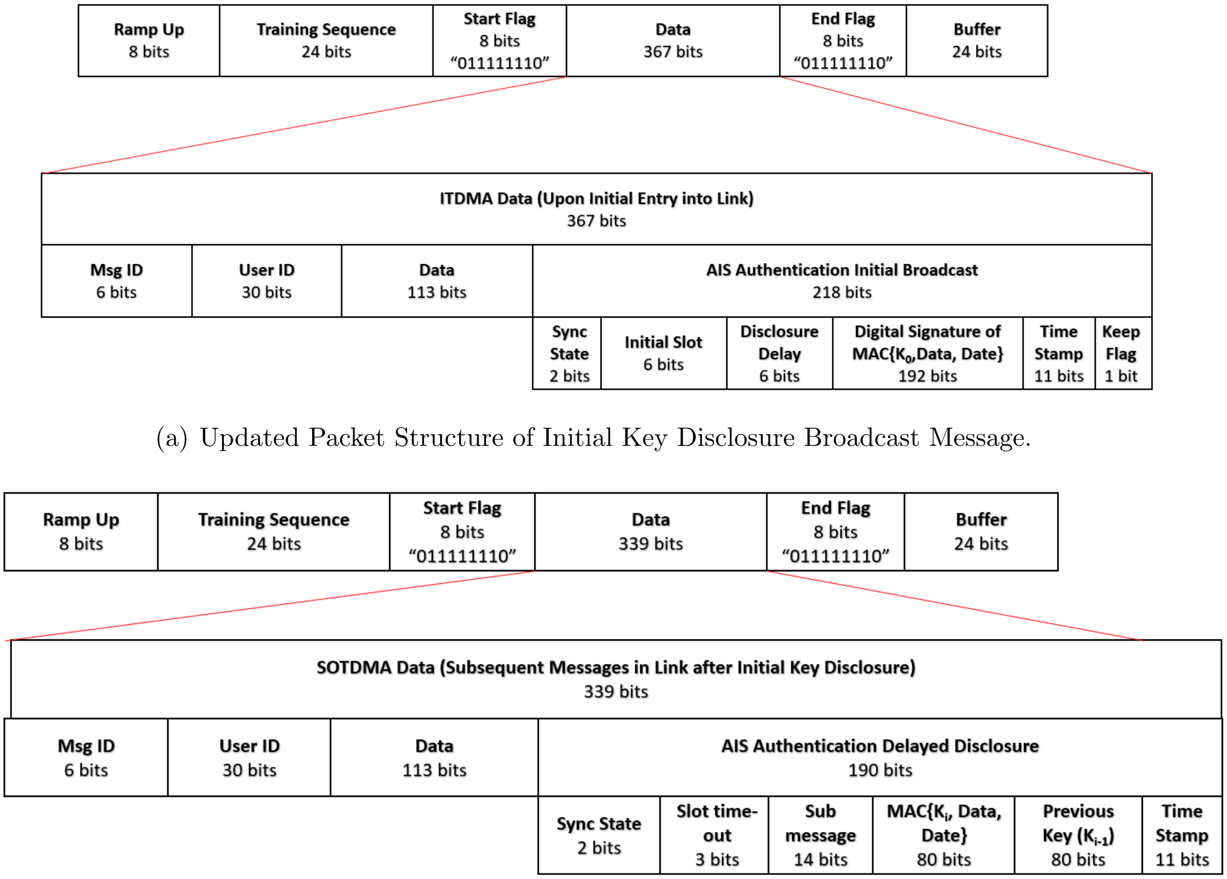
```
Input data:
    • $V$: Random value to begin hash chain seed
    • $N$: Value to represent length of hash chain, $N = (RR * 60) + 6$
    • $T_0$: Time stamp for initial message
    • $A$: AIS Message Data
    • $H$: Hash function
Create blank list, $L$, of size $N + 1$
Set $V$ as first item in $L$
for Each value in $L$ do
    Compute the hash of the previous value in $L$
    Store keys for release in reverse order of creation, $K_0 ... K_{n+1}$
end for
Compute digital signature of first MAC, $S_{-}[MAC(K_0, A, Date)]$
```

The AIS transmitter and receiver were implemented using a USRP B200 SDR and an RTL-SDR, respectively, and the packet structure was modified to incorporate the authentication information as shown in Fig. 24.

Specifically, Fig. 24(a) illustrates the modifications required on the AIS packet structure for an initial broadcast upon powering up an AIS receiver or after the user has exhausted all keys within its key chain and/or needs to re-establish its settings such as reporting rate change. Although the data portion of this message is approximately double the size of a traditional AIS frame, the total message size is only 439 bits and would only need to be broadcast six times per hour. Additionally, this message can be sent within 2 consecutive time slots, which is less than the maximum of 5 consecutive slots listed in the ITU standard.

Algorithm 2 – AIS Transmitter: Broadcasting Authentication Information

```
Input data:
    • $MAC(K_i, A, Date)$
    • $T_i$ = Message Timestamp
if Message is ITDMA then
    Insert $S\_[MAC(K_0, A, Date)], MAC(K_0, A, Date), T_0$ into ITDMA formatted AIS packet
else
    Insert $MAC(K_i, A, Date), K_{i-1}, T_i$ into SOTDMA formatted AIS packet
end if
Broadcast to all stations
```



(a) Updated Packet Structure of Initial Key Disclosure Broadcast Message.

(b) Updated Packet Structure of Subsequent Delayed Key Disclosure Broadcast Message.

Fig. 24: The structure of AIS packets implementing the proposed authentication protocol.

Algorithm 3 – AIS Receiver: Authentication Verification

```
Input data:
    • $S_{-}[MAC(K_0, A, Date)]$ (if ITDMA)
    • $MAC(K_0, A, Date)$
    • $K_{i-1}$, Buffered Key
    • $T_{i-1}$, Time stamp of sent message
    • $RI$, Sender Reporting Interval (derived from sender status/speed/course rate-of-change)
    • $MessageType$, AIS Message Type 1-27 (legacy) or 28+ (AIS Authenticated Message)
if Message Type $\geq 28$ then
    if Message is ITDMA then
        Use sender's public key to validate digital signature and store, $S_{+}[[S_{-}[MAC(K_0, A, Date)]]$
        Store $MAC(K_0, A, Date)$
    else
        if $T_{i-1} > (RI + .1 * NI * .02667)$ then
            USER IS UNKNOWN (Message arrived outside time interval)
        else
            Compute $H(K_{i-1}, A_{i-1}, Date)$, compare to stored MAC
            if Stored MAC and computed MAC are equivalent and Valid Digital Signature on file for this hash chain then
                USER IS TRUSTED AND VERIFIED
            else
                if Stored MAC and computed MAC are equivalent but no valid Digital Signature on file then
                    USER IS VERIFIED
                end if
            end if
        end if
    end if
else
    USER IS UNKNOWN
end if
```

TABLE 1: Comparison of AIS Security Protocols

Method	Data Overhead (bits)	Consecutive TDMA Frames	Cryptography	Security Type
Proposed Authentication Protocol (using TESLA)	203 (initial & digitally signed) 171 subsequent	1.5-2	Asymmetric	ECDSA (NIST-192)
Secure AIS w/ IDBE [30]	331, 672, or 768 (depending on security type)	3+	Asymmetric	SS, MNT
SecureAIS – Securing Pairwise Vessel Communications [28]	Not stated; 880 (estimated)	10 (5 per transceiver)	Symmetric	ECQV/ECDH
pAIS (Suggested in [33])	258	2	Asymmetric	RSA
X.509 Certificates (Suggested in [25], [28])	8000+ (estimated)	85	Asymmetric	X.509

A BLAKE2 hash function was used in the implementation of the TESLA-based authentication protocol, which corresponds to a lightweight hash algorithm that provides more efficient and secure hash generation than SHA-3 and RSA [50]. 80-bit hash values were used as this will provide more than adequate security in the extremely short duration (between 2 and 30 s) during which each chain will be active. The size of each value in the hash chain may be reduced pending further analysis.

Fig. 24(b) shows the structure of subsequent frames that are sent out by a user after their initial key disclosure broadcast. These messages do not require the 192 bit overhead of a digital signature and instead only need to include the MAC, key disclosure of the previous frame, and time stamp. This message is a total of 411 bits and can be sent out in 1.5 frames.

Finally, Table 1 shows a comparison of our results with those developed in other works. [29] provided no packet or security analysis, while authors in [25] only suggested a X.509 certificate based system, which is compared using data derived from [28].

To compare the efficiency of the proposed authentication approach I looked at the additional overhead required for implementation and compared it to that of the alternative approaches in [25, 30, 28, 29, 33]. The comparison is summarized in Table 1. The proposed authentication approach using TESLA requires 76.9% – 80% less data overhead and 80% less consecutive TDMA frames than that in [28]. Relative to [30], the proposed approach requires 38.7% less overhead data for the initial digitally signed message and 48.3% less data for all subsequent messages. Furthermore, the proposed approach uses 1 less TDMA frame as the digital signature is only required to be sent once per hash chain (every hour), which implies that sending 120 messages in an hour would require 48.25% less overhead than [30]. However, even sending out a digital signature every 6 minutes alongside static information in Message Type 5 would result in an additional 1218 bits per hour of data, yet it would still require 42.2% less overhead than [30]. This also provides 25.9% less overhead per hour than pAIS in [33]. The most powerful ideas here are the fact that frequent messages can be sent out without the need to include a digital signature on every one, and there is no need to conduct an initial symmetric key exchange for use in the MAC which frees up slots in the TDMA scheme. This method provides an optimal balance between sending a lengthy digital signature on every message while still allowing new vessels to verify a user's identity even if they were not present during the initial signature broadcast. Coupled with the loss tolerance of the TESLA key derivation process, this system allows for a robust broadcast protocol ideally suited for enhancing the security of the AIS system.

5.5 INTERFACE WITH EXISTING NAVIGATION SYSTEMS

Since TESLA AIS Authentication can be implemented as a software-only update to handle new AIS messages, vessels would only need to include an additional software package to help handle the arrival of updated packets. Specifically, the existing AIS software would handle physical layer data conversion (GMSK demodulation and NRZI decoding) and then the TESLA software would handle link layer functions if new packet types were detected. If old packet types were detected (indicating a user has not yet updated to TESLA authentication), the existing AIS software would function as it normally does, essentially bypassing the TESLA protocols. The only TESLA interaction would be updating a vessel's authentication status on the chart plotter/RADAR as "unknown". As shown earlier in Fig. 19, the authentication protocol will output a notification to the user's chart plotter and radar indicating the trust level of the data, similar to the way one can currently query an AIS target to review its voyage and other static data. Data output from TESLA must adhere to standard NEMA structure for compatibility with ECDIS chart plotters and RADAR systems.

CHAPTER 6

CONCLUSIONS

In this paper I have conducted an extensive review of the AIS system, which I have shown contains significant vulnerabilities that can be easily exploited by an adversary. I provided an extensive review of the existing research on AIS security solutions and explored security research in similar fields including the aviation and ah-hoc vehicular network fields. After careful consideration, I determined that AIS requires both authentication and message integrity and built upon work first theorized in the aviation industry to use TESLA as a broadcast authentication protocol. After this research, I presented my first contribution which is a SDR AIS transmitter and receiver that can be used as a robust test platform for AIS research. Next, I presented my second contribution which was a novel authentication algorithm for the AIS system based on the TESLA protocol that enables receivers of multicast communications to verify the source and integrity of received data packets. Unlike the alternative approaches proposed for authentication in the AIS, the approach presented in this paper can authenticate users without the use of an a priori shared secret key or the need to conduct key exchanges over several messages. This solution requires significantly less overhead than previous solutions and is backward compatible with existing hardware. Additionally, this solution contains an integrated message time stamp that prevents the possibility of a replay attack.

6.1 FUTURE WORK

Future work will include testing data usage of this system when used in conjunction with existing AIS transceivers as well as analyzing the scaling of this system and modeling how it will react in congested waterways where many users are active. Additionally, the inclusion of the PKI signature on the hash chain is cumbersome and requires world-wide agreement on an established CA and private key exchanges. However, there must be a way to initially validate the identity of a user producing a hash chain, so this element of the protocol cannot be ignored. Therefore, more research should be conducted to provide a decentralized and lightweight method to deal with this issue. Potential solutions include using blockchain technology to share keys, or by implementing a certificate-less web-of-trust element in place of a traditional digital signature. Another feature worth exploring is how small you can make the key size while still maintaining adequate security of the hash chain, given that the proposed disclosure rate is on the order of seconds. Finally, work must also be done to ensure the output of this system is seamlessly displayed on maritime chart plotters and radar systems since the data this feature provides is ultimately a tool for the use of shipboard personnel in evaluating safety of navigation decisions.

The proposed authentication scheme does not solve all of the vulnerabilities within the AIS system. As long as the information on an AIS transceiver is configurable by the user, there remains the possibility that PKI and hash chain verified data is still inaccurate. Mariners operating vessels receive extensive training and are required by international law to use all of their available tools to determine risk of collision with another vessel, so AIS is not their sole source of information. Additionally, this authentication protocol inherently relies upon an accurate and available GPS source for both timing and position data. As pointed out in [51], GPS jamming and spoofing are relatively easy, so this represents a vulnerability within the protocol. This is especially true if using rapid reporting rates where time only needs to be shifted slightly so that a user is unaware, eventually leading to lost positional data due to the TESLA system invalidating packets wthat arrived after the set key disclosure schedule (or shifting time altogether to the advantage of an adversary). The vulnerabilities with AIS message 21 (ATON Report) and message 22/23 (group assignment command) must also include verification that they originated only from a competent maritime authority such as the U.S. Coast Guard, not just any valid public key within the system. This can be done through the TESLA authentication protocol but would need to include additional software which would specifically check for these message types and verify that a specific key derived from a maritime authority exists. This additional hierarchical PKI can still originate from the ITU but should be separate from the one used to generate normal key pairs and should only be distributed to the agencies in each country responsible for ATON and AIS. This authentication protocol also relies on the ability of each vessel to securely retrieve their own private key from a CA and also to keep that private key secret. Since this protocol was not implemented using IBE, vessel's must also be able to retrieve the public key of any vessel they interact with on the ocean. This is a simple task with the internet, where a public database can be generated that allows vessels to search for public keys based on any vessel's unique MMSI number. The CA and intermediate authorities would be responsible for ensuring that the MMSI is bound to the correct public key. While larger vessels operating far from shore can use onboard internet capabilities to connect to this database, there is still a problem for smaller vessels. Recreational vessels and other small craft often do not have these capabilities and would therefore need to download the latest copy of the public key database before departing or connect using a cellular connection, although this is unreliable off shore. An IBE scheme for public key retrieval may be one way to alleviate this problem since a vessel's public key would be tied to their static AIS data, although this would require additional message overhead as shown in [30] and [43]. These proposed areas of research will ultimately lead to a more secure AIS system capable of withstanding current cybersecurity threats while allowing public access to vital safety-of-navigation information.

REFERENCES

[1] ITU-R, “Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile band,” *Rec ITU-R M.1371-5*, 2014. [Online]. Available: https://www.itu.int/dms_pubrec/itu-r/rec/m/R-REC-M.1371-5-201402-I!!PDF-E.pdf

[2] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” 2009. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[3] United States Coast Guard, “Oil Pollution Act of 1990 (OPA).” [Online]. Available: https://www.uscg.mil/Mariners/National-Pollution-Funds-Center/About_NPFC/opa/

[4] ITU-R, “Long range detection of automatic identification system (AIS) messages under various tropospheric propagation conditions,” *Report ITU-R M.2123-0*, 2007. [Online]. Available: https://www.itu.int/dms_pub/itu-r/opb/rep/R-REP-M.2123-2007-PDF-E.pdf

[5] L. Luft, I. Gonin, and D. Pietraszewski, “Researching Technology Improvements for AIS,” U.S. Coast Guard Research and Development Center (RDC), New London, CT, Tech. Rep. July, 2018.

[6] International Maritime Organization, “Safety of Life at Sea - Safety of Navigation Chapter V,” *SOLAS Convention*, p. 29, 2002. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/343175/solas_v_on_safety_of_navigation.pdf

[7] United States Code, “Automatic Identification System.” [Online]. Available: https://ecfr.federalregister.gov/current/title-33/chapter-I/subchapter-P/part-164

[8] International Maritime Organization, “Guidelines on annual Testing of the Automatic Identification System (AIS),” Tech. Rep., 2007.

[9] United States Coast Guard, “USCG AIS Inspection Checklist & Report,” pp. 4–7, 2019. [Online]. Available: https://www.navcen.uscg.gov/pdf/AIS/USCG_AIS_Inspection_Checklist19_10_01.pdf

[10] United States Coast Guard, "Automatic Identification System Encoding Guide," no. Iso 3166, 2012. [Online]. Available: http://www.uscg.mil/hq/cg5/TVNCOE/Documents/links/AIS.EncodingGuide.pdf

[11] United States Coast Guard NAVCEN, "AIS Frequently Asked Questions." [Online]. Available: https://www.navcen.uscg.gov/?pageName=AISFAQ#5

[12] United States Coast Guard NAVCEN, "AIS Problem Report." [Online]. Available: https://www.navcen.uscg.gov/?pageName=AISProblem

[13] United States Coast Guard NAVCEN, "Vessel Information Verification Service."

[14] United States Coast Guard, "Automatic Identification System (AIS) – Accurate Broadcasts Don't Happen Automatically," Tech. Rep., 2020. [Online]. Available: https://www.dco.uscg.mil/Portals/9/DCODocuments/5p/CG-5PC/INV/Alerts/USCGSA_0420.pdf?ver=2020-05-13-090105-050

[15] "NMEA 0183." [Online]. Available: https://www.nmea.org/content/STANDARDS/NMEA_0183_Standard

[16] J. G. Proakis and M. Salehi, *Communication Systems Engineering*, 2002.

[17] T. Turletti, "GMSK in a nutshell," pp. 1–6, 1996.

[18] ITU-R, "Assignment and use of identities in the maritime mobile service," *Rec. ITU-R M.585-7*, vol. 8, 2015.

[19] ITU-R, "Table of Maritime Identification Digits." [Online]. Available: https://www.itu.int/en/ITU-R/terrestrial/fmd/Pages/mid.aspx

[20] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach*, 7th ed., 2016.

[21] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 196 LNCS, pp. 47–53, 1985.

[22] R. J. Reisman, "Air traffic management blockchain infrastructure for security, authentication, and privacy," *AIAA Scitech 2019 Forum*, pp. 1–14, 2019.

[23] P. R. Zimmerman, *The Official PGP User's Guide.* Cambridge, MA, USA: MIT Press, 1995.

[24] G. Guo, J. Zhang, and J. Vassileva, "Improving PGP Web of Trust through the expansion of trusted neighborhood," in *2011 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 1. IEEE, 2011, pp. 489–494.

[25] M. Balduzzi, A. Pasta, and K. Wilhoit, "A security evaluation of AIS automated identification system," *ACM International Conference Proceeding Series*, vol. 2014-Decem, no. December, pp. 436–445, 2014.

[26] United States Coast Guard, "CAUTION TO AIS USERS: YOU MAY BE INADVERTENTLY OPERATING ON DIFFERENT AIS CHANNELS," Tech. Rep., 2010. [Online]. Available: https://www.navcen.uscg.gov/pdf/AIS/USCG_Safety_Alert_07_10_Excerpts.pdf

[27] S. Jayasimha, J. Paladugula, A. V. Gadiraju, and M. K. Medam, "Satellite-based AIS receiver for dense maritime zones," *2017 9th International Conference on Communication Systems and Networks, COMSNETS 2017*, pp. 15–22, 2017.

[28] A. Aziz, P. Tedeschi, S. Sciancalepore, and R. D. Pietro, "SecureAIS - Securing Pairwise Vessels Communications," in *2020 IEEE Conference on Communications and Network Security, CNS 2020*, 2020.

[29] J. Hall, J. Lee, J. Benin, C. Armstrong, and H. Owen, "IEEE 1609 influenced automatic identification system (AIS)," *IEEE Vehicular Technology Conference*, vol. 2015, pp. 0–4, 2015.

[30] A. Goudosis and S. K. Katsikas, "Secure ais with identity-based authentication and encryption," *TransNav*, vol. 14, no. 2, pp. 287–298, 2020.

[31] M. W. Parsons, "Encrypted Automatic Identification System (EAIS) Interface Design Description (IDD)," Tech. Rep. June, 2014. [Online]. Available: https://epic.org/foia/dhs/uscg/nais/EPIC-15-05-29-USCG-FOIA-20151030-Production-2.pdf

[32] National Institute of Standards and Technology, "Federal Information Processing Standards Publication 197," Tech. Rep., 2001.

[33] G. C. Kessler, "Protected ais: A demonstration of capability scheme to provide authentication and message integrity," *TransNav*, vol. 14, no. 2, pp. 279–286, 2020.

[34] M. Strohmeier, V. Lenders, and I. Martinovic, “On the security of the automatic dependent surveillance-broadcast protocol,” *IEEE Communications Surveys and Tutorials*, vol. 17, no. 2, pp. 1066–1087, 2015.

[35] A. Perrig, R. Canetti, J. Tygar, and D. Song, “Efficient authentication and signing of multicast streams over lossy channels,” *Proceedings 2000 IEEE Symposium on Security and Privacy*, pp. 56–73, 2000.

[36] A. Perrig, R. Canetti, J. D. Tygar, and D. Song, “The TESLA Broadcast Authentication Protocol,” *CryptoBytes*, vol. 5, no. 2, pp. 2–13, Summer/Fall 2002, published by RSA Laboratories.

[37] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, “SPINS: Security protocols for sensor networks,” *ACM Journal of Wireless Networks*, vol. 8, no. 5, pp. 521–534, 2002.

[38] A. Perrig, R. Canetti, J. Tygar, and D. Song, “Efficient and Secure Source Authentication for Multicast,” in *Proceedings of the Internet Society 2001 Network and Distributed System Security Symposium (NDSS)*, 2001.

[39] D. Liu and P. Ning, “Multi-Level microTESLA: A Broadcast Authentication System for Distributed Sensor Networks,” *ACM Transactions on Embedded Computing Systems*, vol. 3, no. 4, pp. 800–836, 2003.

[40] Y.-c. Hu and K. P. Laberteaux, “Strong VANET SEcurity on a Budget,” in *Embedded Security in Cars (ESCAR)*, vol. 06, 2006, pp. 1–9.

[41] S. Sciancalepore and R. Di Pietro, “SOS: Standard-Compliant and Packet Loss Tolerant Security Framework for ADS-B Communications,” *IEEE Transactions on Dependable and Secure Computing*, vol. 5971, no. c, pp. 1–18, 2019.

[42] S. Capkun, L. Buttyán, and J. P. Hubaux, “Self-organized public-key management for mobile ad hoc networks,” *IEEE Transactions on Mobile Computing*, vol. 2, no. 1, pp. 52–64, 2003.

[43] K. Hamouid and K. Adi, “Efficient certificateless web-of-trust model for public-key authentication in MANET,” *Computer Communications*, vol. 63, no. 1, pp. 24–39, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2015.02.009

[44] J. Mitola, “Software radios-survey, critical evaluation and future directions,” in *NTC-92: National Telesystems Conference*, 1992, pp. 13/15–13/23.

[45] B. Stewart, K. Barlee, D. Atkinson, and L. Crockett, *Software Defined Radio Workflow Using MATLAB & Simulink and the RTL-SDR*, 2015. [Online]. Available: https://www.desktopsdr.com/

[46] E. Research, “N. I. B. Ettus Research, “USRP B200”.” [Online]. Available: https://www.ettus.com/all-products/ub200-kit/

[47] Nooelec, “Nooelec NESDR SMArt v4 SDR - Premium RTL-SDR w/ Aluminum Enclosure, 0.5PPM TCXO, SMA Input. RTL2832U & R820T2-Based.” [Online]. Available: https://www.nooelec.com/store/sdr/sdr-receivers/nesdr-smart-sdr.html

[48] G. Radio, “GNU Radio Wikipedia.” [Online]. Available: https://wiki.gnuradio.org/index.php/Main_Page

[49] “OpenCPN GitHub.” [Online]. Available: https://github.com/OpenCPN/OpenCPNhttps://github.com/OpenCPN/OpenCPN

[50] V. Rao and K. V. Prema, “Comparative Study of Lightweight Hashing Functions for Resource Constrained Devices of IoT,” *CSITSS 2019 - 2019 4th International Conference on Computational Systems and Information Technology for Sustainable Solution, Proceedings*, pp. 0–4, 2019.

[51] G. C. Kessler and S. D. Shephard, *Maritime Cybersecurity: A Guide for Leaders and Managers*, 2020.

APPENDIX A

MATLAB CODE

```
close all
clear all
clc

Fs = 2000;
hss = dsp.SpectrumAnalyzer('SampleRate', Fs);
data = randi([0 1],3000,1);%I had to increase the # of points
%% OQPSK Modulate data
hMod = comm.OQPSKModulator('BitInput',true);
OQPSK = step(hMod, data);
hMod2 = comm.GMSKModulator('BitInput',true, 'BandwidthTimeProduct', .5);
GMSK=step(hMod2,data);
hMod3 = comm.QPSKModulator('BitInput',true);
QPSK = step(hMod3, data);
hMod4 = comm.MSKModulator('BitInput', true);
MSK=step(hMod4,data);

hMod5=comm.FSKModulator('BitInput', true);
FSK=step(hMod5, data);

% %% Add noise
% %hAWGN = comm.AWGNChannel('EbNo',2);
% rx = step(hMod, tx);
%
% %This is the line that makes it work, passing in a matrix of input data
% step(hss,[rx tx]);
r = angle(MSK);
d = 1:1:length(MSK);

r1=angle(FSK);
d1=1:1:length(FSK);
```

```
r2=angle(GMSK);
d2=1:1:length(GMSK);

div = 1000;
figure(3)
plot(d(1:div), r(1:div));
hold on
plot(d1(1:div),r1(1:div));
legend ('MSK', 'FSK')
%title('Phase Comparison of CPFSK and FSK')
xlabel('Sample')
ylabel('Phase (rad)')

figure(4)
plot(d(1:div/2), r(1:div/2));
hold on
plot(d2(1:div/2),r2(1:div/2));
legend ('MSK', 'GMSK')
%title('Phase Comparison of MSK and GMSK')
xlabel('Sample')
ylabel('Phase (rad)')

[pxx, w1] = pwelch(OQPSK);
[pxy, w2] = pwelch(GMSK);
[pyy, w3] = pwelch(QPSK);
[pyx, w4] = pwelch(MSK);
figure(1)
plot(w1, 10*log10(pxx), 'r');
hold on
plot(w2, 10*log10(pxy), 'b');
hold on
plot(w3, 10*log10(pyy), 'm');
hold on
plot(w4, 10*log10(pyx), 'g');

ylabel('Power Spectral Density (dB/(rad/sample))');
xlabel('Normalized Frequency');
legend('OQPSK', 'GMSK', 'QPSK', 'MSK');
%title('Power Spectral Density of Modulation Schemes');

figure(2)
```

```
Gmod1 = comm.GMSKModulator('BitInput',true, 'BandwidthTimeProduct', .1);
Gmod2 = comm.GMSKModulator('BitInput',true, 'BandwidthTimeProduct', .2);
Gmod3 = comm.GMSKModulator('BitInput',true, 'BandwidthTimeProduct', .3);
Gmod4 = comm.GMSKModulator('BitInput',true, 'BandwidthTimeProduct', .4);
Gmod5 = comm.GMSKModulator('BitInput',true, 'BandwidthTimeProduct', .5);
Gmod6 = comm.GMSKModulator('BitInput',true, 'BandwidthTimeProduct', 1);
GMSK1=step(Gmod1,data);
GMSK2=step(Gmod2,data);
GMSK3=step(Gmod3,data);
GMSK4=step(Gmod4,data);
GMSK5=step(Gmod5,data);
GMSK6=step(Gmod6,data);
[px1, x1] = pwelch(GMSK1);
[px2, x2] = pwelch(GMSK2);
[px3, x3] = pwelch(GMSK3);
[px4, x4] = pwelch(GMSK4);
[px5, x5] = pwelch(GMSK5);
[px6, x6] = pwelch(GMSK6);

plot(x1, 10*log10(px1));
hold on
plot(x2, 10*log10(px2));
hold on
plot(x3, 10*log10(px3));
hold on
plot(x4, 10*log10(px4));
hold on
plot(x5, 10*log10(px5));
hold on
plot(x6, 10*log10(px6));
legend('.1', '.2','.3', '.4', '.5', '1');
ylabel('Power Spectral Density (dB/(rad/sample))');
xlabel('Normalized Frequency');
%title('Time Bandwidth Product GMSK Modulation');
```

VITA

Robert E. Litts
Department of Electrical and Computer Engineering
Old Dominion University
Norfolk, VA 23529

Education

- M.S. Electrical and Computer Engineering, May 2021, Old Dominion University
- B.S. Electrical and Computer Engineering, May 2013, United States Coast Guard Academy

Select Publications

- R. Litts, D. Popescu, O. Popescu, "Authentication Protocol for Enhanced Security of the Automatic Identification System," submitted to *IEEE International Black Sea Conference on Communications and Networking*, Apr 2021.

Typeset using LaTeX.