SecureAIS - Securing Pairwise Vessels Communications

Ahmed Aziz, Pietro Tedeschi, Savio Sciancalepore, Roberto Di Pietro
Division of Information and Computing Technology (ICT)
College of Science and Engineering (CSE), Hamad Bin Khalifa University (HBKU), Doha, Qatar
{aaziz, ptedeschi}@mail.hbku.edu.qa, {ssciancalepore, rdipietro}@hbku.edu.qa

Abstract— **Modern vessels increasingly rely on the Automatic Identification System (AIS) digital technology to wirelessly broadcast identification and position information to neighboring vessels and ports. AIS is a time-slotted protocol that also provides unicast messages—usually employed to manage self-separation and to exchange safety information. However, AIS does not provide any security feature. The messages are exchanged in clear-text, and they are not authenticated, being vulnerable to several attacks, including forging and replay. Despite the existing contributions in the literature propose some strategies to overcome the exposed weaknesses, some are insecure, others do not comply with the standard, while the remaining few ones would introduce an unacceptable overhead—that is, they would require a relevant number of AIS-time slots, because of the limited payload available for each time-slot.**

In this paper, we propose *SecureAIS*, a key agreement scheme that allows any pair of vessels in reach of an AIS radio to agree on a shared session key of the desired length, by requiring just a fraction of the AIS time-slots of competing solutions. Further, the scheme is fully standard compliant and does not require any modification to the available hardware. The proposed scheme integrates the Elliptic Curve Qu-Vanstone (ECQV) implicit certification scheme and the Elliptic Curve Diffie Hellman (ECDH) key agreement technique, requiring moderate computational overhead while enjoying an optimal usage of the available bandwidth. When configured with the highest security level of 256 bits, *SecureAIS* allows two AIS transceivers to agree on a shared session key in 20 time-slots, against the 96 time-slots required by the competing solution based on traditional X.509 certificates. The proposed solution has been implemented and tested in a real scenario, while its security has been formally evaluated through the ProVerif tool. Finally, the source code of our Proof-of-concept using GNURadio and Ettus Research X310 has been also released as open-source to pave the way to further research by both Industry and Academia in maritime communication security.

Index Terms—**AIS, Vessels Cybersecurity, Mutual Authentication, Cyber-Physical Systems Security, ECQV**

I. Introduction

Automatic Identification System (AIS) is the standard communication technology used by vessels to broadcast information about their position and identification data, as well as to communicate with other vessels or AIS transceivers located on the ground [1]. Starting from 2002, the usage of AIS has become mandatory on-board of all passenger ships and commercial vessels exceeding the 300 tonnes [2]. Today, AIS is used by over $650,000$ vessels throughout the world, and more than 18 million AIS messages are exchanged and recorded on a monthly basis [3].

Unfortunately, its design dating back to the early 1990s, AIS does not include any security feature. Indeed, messages are transmitted in clear-text, and are not authenticated either. Thus, adversaries equipped with compatible AIS transceivers or cheap Software Defined Radios (SDRs) can easily generate rogue AIS messages, that cannot be distinguished from legitimate ones. These weaknesses enable several attacks, including replay and impersonation, to name a few. In turn, these attacks can create safety issues, e.g., forcing the vessels to change their route or preventing ships to rescue distressed vessels [4].

Despite these weaknesses have been already identified by some contributions in the literature, only few of them provide solutions to address security issues of the AIS communication technology. The majority of the cited solutions focus either on broadcast authentication [5] or anomaly detection, identifying on the servers-side the emission of rogue AIS messages [6], [7]. Concerning the insecurity of vessel-to-vessel and vessel-to-ground unicast communication links, despite the availability of several solutions outside of the AIS domain, to the best of our knowledge, only two contributions provided possible solutions specifically tailored to AIS. Specifically, the authors in [4] proposed the usage of key agreement schemes based on traditional X.509 certificates, while the authors in [8] discussed the design of a security architecture based on Identity Based Cryptography (IBC) techniques. However, both proposals have not been implemented, and indeed have issues when transferred into the AIS communication technology. While the protocol in [8] is not authenticated (leading to Impersonation and Denial-of-Service attacks), the usage of X.509 certificates does not consider the bandwidth constraints of AIS. Indeed, AIS is characterized by very short messages (only 256 bits, headers included), and thus, the integration of traditional certificates would result in an overwhelming message overhead and bandwidth (slots) consumption. Furthermore, the above techniques neither integrated the proposed security solution within the AIS standard, nor they implemented the proposed schemes within experimental systems using AIS.

Contribution. To overcome the gaps shown before, in this paper we present *SecureAIS*, a key establishment protocol specifically designed to leverage the features and meet the bandwidth constraints of the AIS communication technology. *SecureAIS* has been designed as a software-only, standard-compliant AIS application, that can be installed by Class-A and Class-B AIS transceivers to establish a pairwise key. This key can be used to encrypt and authenticate messages exchanged between the two AIS transceivers. *SecureAIS* integrates solid cryptographic blocks, such as the Elliptic Curve Qu-Vanstone (ECQV) implicit certification scheme and the Elliptic Curve Diffie-Hellman (ECDH) key agreement algorithm. Further, its security has been formally verified using the automated tool ProVerif. Moreover, a real proof-of-concept of *SecureAIS* has been implemented, using Ettus Research X310 SDRs and GNURadio development toolkit. Our results show that considering the maximum security level of 256 bits, *SecureAIS* allows two AIS transceivers to establish a secure session key in only 20 AIS time-slots. In our Proof-of-Concept using GNURadio, this maps to 3.164 s, where most of the overhead (the 99%) is due to the specific system setup overhead incurred when using GNURadio—that is, when deployed on commercial radios, our solution would require less than 40 ms to establish a symmetric key. The comparison with X.509 certificates shows the bandwidth efficiency achieved via our solution, reducing the overhead of the 79%. To the best of our knowledge, *SecureAIS* is the first key agreement scheme specifically designed as a standard-compliant solution for the AIS communication technology, and that can be integrated into modern transceivers as a simple software update. The code of *SecureAIS* and the related formal security verification have been released as open-source [9], to allow interested readers to verify our claims and further boost the research and implementation of security solutions for the AIS technology.

Roadmap. The rest of this contribution is organized as follows. Section II provides the background on the technologies and schemes used in this paper; Section III describes the scenario and the adversarial model, while Section IV details the *SecureAIS* scheme. The security analysis of the protocol has been provided in Section V, while the experimental tests using a real proof-of-concept have been reported in Section VI. Finally, Section VII wraps-up the paper and draws future research directions.

II. Background

In this section, we provide the background details concerning the communication technology and tools discussed throughout this paper. Specifically, Section II-A introduces the AIS technology, while Section II-B delves into the security aspects of AIS. Finally, Section II-C introduces the ECQV implicit certification scheme adopted within the *SecureAIS* protocol.

A. *Automatic Identification System (AIS)*

The AIS communication technology has been developed in the early 1990s, to assist vessels in communicating with nearby ships and port operators. Since the year 2002,

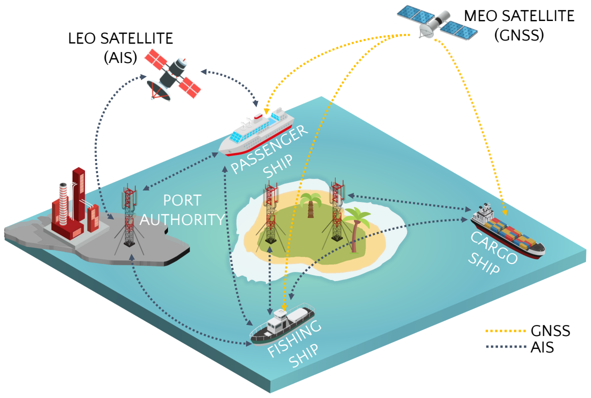


Figure 1. Overview of the AIS communication architecture.

the International Maritime Organization (IMO) mandated its adoption and usage on-board of all commercial ships exceeding a gross tonnage of 300 tonnes, as well as all the passenger vessels, irrespective of their weight [4].

AIS is used for many services. First, it is used regularly by any vessel to broadcast identification and position data, acquired via regular Global Navigation Satellite System (GNSS) technologies. This feature enables several applications, such as identification, route adjustment, accident prevention and investigation, search and rescue operations, and remote tracking, to name a few. Besides, two vessels can establish a one-to-one communication, exchanging dedicated binary messages. We notice that some AIS transceivers are also mounted within ports and along the coastal tracks, to enhance ship tracking and assist in search-and-rescue operations.

At the physical-layer, AIS operates in the Very High Frequency (VHF) frequency band, leveraging two main channels, 161.975 MHz and 162.025 MHz. Over these carriers, AIS uses the Gaussian filtered Minimum Shift Keying (GMSK) modulation with a bit-rate of 9600 bit/s. In regular operational conditions, the transmission range of the *terrestrial* AIS technology can reach a maximum theoretical distance of about 70 km, even if practical conditions and weather factors often reduce its range to approximately 40 km. Since 2005, AIS transceivers have been installed on Low-Earth Orbit (LEO) satellites, provided by the ORBCOMM satellite operator. These developments lead to the introduction of the *Space-based AIS* (SAT-AIS), enhancing the direct coverage of the AIS technology to range up to 400 km. An high-level architectural overview of AIS is depicted in Figure 1, summarizing the above considerations.

The standard format of AIS messages is illustrated in Figure 2. AIS messages have an overall size of 256 bits [10]. The message starts with the first 8 bits reserved to turn on the AIS transceiver. Then, the message contains a preamble of 24 bits, used to identify an upcoming AIS message, and for the receiver to synchronize with the symbols emitted by the transmitter. A *Start Flag* of 8 bits denotes the start of the AIS data message, having a fixed size of 168 bits only. The bottom part of Figure 2 shows the content of AIS

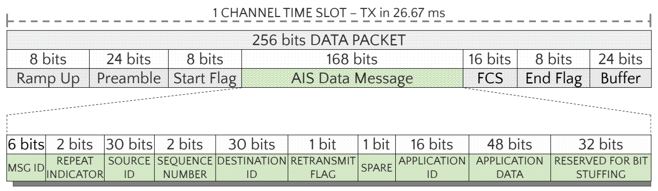


Figure 2. Format of AIS messages: Generic AIS frame (upper part) and detail of the content of AIS type 6 binary messages (bottom part).

binary messages (Message Type 6), being these messages the ones specifically considered in this work. They contain the *Message Type* (6 bits) to indicate the type of the message, the *Repeat Indicator* (2 bits) to specify if the message should be rebroadcast, the *Source ID* Maritime Mobile Service Identity (MMSI) (30 bits), uniquely identifying the transceiver emitting the message, the Sequence Number (2 bits) of the packet (used for sequencing), the *Destination ID*, MMSI (30 bits) of the intended destination, a *Retransmit Flag* (1 bit) used to indicate if this packet is re-transmitted, a *spare bit* (1 bit) reserved for future use, the *Application ID* of 16 bits, specifying the unique identifier of the application, the application data of 48 bits to containing binary data, and the last 32 bits devoted to bit stuffing. The *Frame Check Sequence (FCS)* field follows the AIS data message and it is used to provide error detection, via a Cyclic Redundancy Check (CRC) polynomial of 16 bits. The *End Flag* byte closes the transmission of the message. Finally, the *Buffer* field of 24 bits should be adopted only for *bit-stuffing*, *distance delay* and *synchronization jitter*. Overall, the time from the transmission of the first bit of the preamble to the end flag is always less than 26.667 ms (duration of the slot).

We highlight that AIS RF operations take place according to a slot-based Time Division Multiple Access (TDMA) schedule. Specifically, time is divided into slots, lasting 26.667 ms. Each AIS transceiver periodically acquires the knowledge of the AIS entities in the radio neighborhood, by passively listening to one of the two AIS channels for 1 minute ($2,250$ slots). Based on this knowledge, it selects for transmissions one or more free slots, i.e., time-slots where none RF operation was detected. According to the amount of data to be transmitted, one or more time-slots can be required. The standard mandates that a transceiver can continuously transmit data for a maximum number of consecutive slots, as a continuous data stream, depending on its *class*. Class A devices can transmit continuously for 5 consecutive slots, while transmitting Class B devices can occupy the medium for a maximum of 3 consecutive slots. The overall number of binary data bytes that can be transmitted continuously are summarized in Table I. If the transceiver requires to transmit more data, a new transmission operation could be triggered immediately after the previous one. We refer the interested readers to the AIS standard specification for more details about the channel logic of AIS [10, p. 116].

Table I
MAXIMUM BINARY DATA BYTES THAT COULD BE TRANSMITTED USING A GIVEN NUMBER OF CONSECUTIVE TIME-SLOTS.

Number of Consecutive Slots	Maximum binary data bytes
1	8
2	36
3	64
4	92
5	117

B. Security Considerations

As previously remarked, the AIS protocol does not provide any security service. Indeed, messages transmitted according to the AIS standard are not encrypted, enabling any compatible receiver to detect and decode the messages. Moreover, AIS messages are not authenticated. This weakness enables several active attacks, such as *replay*, *message manipulation*, *fake vessel injection*, and *spoofing*, to name a few.

The lack of security characterizing AIS is due to several factors. First, at the time when AIS was designed, network security and privacy were not system requirements, and they were generally conceived as optional features. Moreover, the lack of encryption enabled an easy and timely tracking of the vessels, particularly useful for search and rescue operations. Furthermore, the technical skills required to launch radio attacks were reputed too hard to be realized in practice, due to the general unavailability of dedicated equipment. However, on the one side the outstanding technological progress of the last decades has led to the wide availability of dedicated equipment, such as the cheap and easy-to-use SDR, allowing anyone with basic skills to launch potentially harmful attacks. On the other side, progress on both the HW and the cryptographic primitives have enabled the addition of a seamless security layer.

C. ECQV Implicit Certificates

ECQV implicit certificates are a lightweight variant of the traditional explicit certification scheme, recently proposed by the authors in [11]. The most important feature of an implicit certificate is the elimination of an explicit signature from the conventional certificate, with a consistent reduction in the number of bits to be transferred to authenticate the public key. An implicit certificate is defined only by some identification data and a cryptographic value, namely the *Implicit Certificate*, which allows any third-party to extract the public key of an entity. This process is executed using the public key of the Certification Authority (CA), thus implicitly providing the authenticity of the public key.

The following Figure 3 depicts the generation of private and public keys through the ECQV scheme.

The ECQV scheme leverages an elliptic curve group E, a generator G 2 E of order Ω of the group E, and a cryptographic hashing function $H( )$. Let us assume that C and c 2 $(0, \Omega)$ are the public and the private keys of the CA, respectively. To request the implicit certificate M_i, a generic device i with identity I_i generates a pseudo-random value

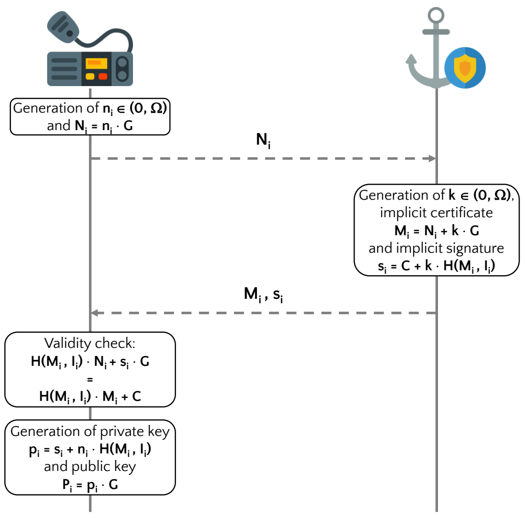

Generation of $n_i \in (0, \Omega)$ and $N_i = n_i \cdot G$
N_i
Generation of $k \in (0, \Omega)$, implicit certificate $M_i = N_i + k \cdot G$ and implicit signature $s_i = C + k \cdot H(M_i, I_i)$
M_i, s_i
Validity check: $H(M_i, I_i) \cdot N_i + s_i \cdot G = H(M_i, I_i) \cdot M_i + C$
Generation of private key $p_i = s_i + n_i \cdot H(M_i, I_i)$ and public key $P_i = p_i \cdot G$

Figure 3. Generation of Private and Public Keys with the ECQV scheme.

n_i 2 $(0, \Omega)$, computes $N_i = n_i \;\; G$, and sends this last value in a request to the CA. On receiving this request, the CA generates a pseudo-random value k 2 $(0, \Omega)$ and computes the implicit certificate M_i and the implicit signature s_i as in the following Eqs. 1 and 2.

$$M_i = N_i + k \;\; G, \tag{1}$$

$$s_i = c + k \;\; H(M_i, I_i). \tag{2}$$

On receiving the implicit certificate M_i and the implicit signature s_i, the device i verifies that the implicit certificate is authentic and released from the trusted CA by verifying the condition reported in Eq. 3.

$$H(M_i jj I_i) \;\; N_i + s_i \;\; G = H(M_i jj I_i) \;\; M_i + C. \tag{3}$$

Then, the device i generates its private key p_i and its public key P_i according to eqs. 4 and 5.

$$p_i = s_i + n_i \;\; H(M_i, I_i), \tag{4}$$

$$P_i = p_i \;\; G. \tag{5}$$

Starting from the knowledge of the implicit certificate M_i, the identity I_i associated with the device, and the public key of the Certification Authority, namely C, any third party can reconstruct the public key P_i of the device i.

$$P_i = p_i \;\; G = C + H(M_i, I_i) \;\; M_i. \tag{6}$$

The derivation of a public key is an operation faster than the verification of a digital signature. This feature makes implicit certificates faster than conventional certificates (e.g. X.509 certificates). Finally, the size of implicit certificates is smaller than the size of explicit certificates.

We remark that ECQV certificates have been already used in the context of the Internet of Things (IoT) networks, to reduce the bandwidth overhead and the energy consumption [12]. However, none previous papers discussed their standard-compliant integration in the AIS technology, neither provided an evaluation of the cost required to support their functionalities in vessel-to-vessel communications.

III. System and Adversary Model

The following section provides the details of the system and the adversary model assumed in this work. Specifically, in Section III-A we illustrate the scenario assumed in our work, while the capabilities and the aim of the adversary are detailed in Section III-B.

A. System Model

In this paper we assume two AIS transceivers, namely A and B, would like to establish a secure communication channel over the AIS communication technology, to exchange confidential information. Our scenario is general, i.e., it can involve either two ships located offshore, far away from the land, or a ship and a port authority, with the ship approaching the port. In both cases, we assume that AIS is the only possible bidirectional communication technology they can use to exchange confidential information.

We assume that the two entities A and B are loosely time-synchronized. This is consistent with a generic maritime scenario, where the AIS transceivers are synchronized to the time provided by one of the many GNSS technologies. Also, we assume that the entities mounting the two transceivers can successfully detect and reject GNSS spoofing attacks. This can be achieved through a variety of solutions, working both at the software level and at the physical layer [13], [14].

We also assume that the two legitimate entities are equipped with *regular* computational and storage capabilities, in line with the capabilities of modern laptops. This is a reasonable assumption, as all the vessels compliant to Safety of Life at Sea (SOLAS) regulations usually integrate dedicated computing systems on-board, providing communication capabilities (AIS, GNSS, SATCOM, to name a few), processing, and visualization services via the Electronic Chart Display and Information Systems (ECDIS) system.

We do not make any assumption on the specific communication architecture of the AIS system. The communication can be either point-to-point through the legacy terrestrial AIS or mediated by a satellite, according to the SAT-AIS communication architecture. In Section V we will show that the security our protocol is not affected by the presence of the proxy and that the protocol is secure even in the unlikely case where the AIS proxy mounted on the satellite is compromised.

Finally, we assume that more than two vessels are located in the same AIS transmission range. Therefore, the identification of the vessel transmitting a given message is not straightforward. We recall that the standard communication range of the terrestrial AIS system can be up to a distance of 70 km, despite the presence of adverse meteorological conditions that could affect direct RF visibility between vessels.

B. Adversary Model

In the following, we assume a powerful attacker, in line with the widely accepted Dolev-Yao model [15]. As such, the adversary assumed in our work is characterized by both passive and active features.

We assume that the adversary is equipped with either an SDR with compatible antennas or an AIS-compliant transceiver. Thus, it only needs to turn on the equipment and tune the radio on the frequency used by the AIS communication technology, to listen to all ongoing AIS communications in the area. Therefore, our adversary is a global passive eavesdropper, able to access to the raw bits of all the messages exchanged between the legitimate parties.

Besides, the adversary can either replay messages received by other devices or transmit its own messages on the communication channel, impersonating one or more AIS transceivers. This can be easily achieved by selecting the same MMSI of a legitimate entity in the area. It can also modify messages on-the-fly, and delete messages from the channel by selectively jamming them.

Using the above-described attack tools, the adversary aims at impersonating one of the vessels in the area and carry out a *Man-In-The-Middle* attack, being accepted as a legitimate communication party and to read all the information directed towards this entity.

IV. The SecureAIS Protocol

The proposed protocol, namely *SecureAIS*, allows two AIS transceivers (being either two vessels or a vessel and a port authority) to establish a pairwise shared key, to be used to secure all the following binary messages exchanged over the AIS communication technology. The actors involved in the *SecureAIS* scheme are introduced in Section IV-A. The *SecureAIS* protocol is organized in two main phases, namely the *Setup* and *Online* phases, thoroughly described in the following Section IV-B and Section IV-C, respectively.

A. Actors of the SecureAIS Scheme

The system considered in this work involves the following actors.

Vessel. It is a generic vessel, equipped with an AIS transceiver, used to emit broadcast messages and initiate unicast communications with other AIS transceivers. It also features regular computational capabilities, equivalent to the ones of a regular medium-end commercial laptop.

Port Authority. It is an AIS transceiver installed within a port, in charge of: (i) monitoring vessels moving throughout the area of the port, and (ii) interact with the vessels to provide the requested information, such as a secure docking site or a dedicated route.

Maritime Authority. It is a Trusted Third-Party, in charge of regulating communication aspects among vessels and port authorities. It is responsible for issuing cryptography materials for vessels and port authorities, including the private and public key pairs of the AIS transceivers, and trusted certificates that certify the unique relationship between the AIS transceiver and the specific entity where the transceiver is installed. It is worth noting that this authority could be not always online and available to the vessels (e.g., it could not be reachable when the vessels are located off-shore). In the real world, this role could be played by the IMO.

The notation used in the paper is summarized in Table II.

Table II
Notation used in the paper.

Notation	Description
c	Private Key of the Maritime Authority
C	Public Key of the Maritime Authority
H	Generic Hashing Function
G	Generator Point of the Elliptic Curve
t	Security Level Indicator
I_i	MMSI of the AIS Transceiver i
p_i	Private Key of the AIS Transceiver i
P_i	Public Key of the AIS Transceiver i
M_i	Implicit Certificate of the AIS Transceiver i
s_i	Implicit Signature of the certificate of the AIS Transceiver i
r_i	Random Nonce generated by the AIS Transceiver i
λ_i	Authentication Proof generated by the AIS Transceiver i
K_i'	Temporary Session Key generated by the AIS Transceiver i
K_i	Final Session Key generated by the AIS Transceiver i

B. Setup Phase

The Setup Phase is executed at the boot time by each AIS transceiver. In this phase, the transceiver is equipped with the cryptography materials used in the key agreement scheme. These include:

The generator point G of the elliptic curve.
The hashing function H.
The Hashed Message Authentication Code (HMAC) function $HMAC$.
The key derivation function KDF.
The public Key of the Maritime Authority, namely C.
The implicit certificate M_i released by the Maritime Authority.
The private and public Key of the transceiver i, namely p_i and P_i, respectively, generated via the ECQV mechanism described in Section II. The interactions executed offline between the manufacturer of the transceiver i and the Maritime Authority are the same as shown in Figure 3, with the difference that the Maritime Authority replaces the identity string I_i (correspondent to the MMSI) with an identification string l_i, generated as in the following Eq. 7.

$$l_i = (I_i jj V_i), \tag{7}$$

where I_i is the MMSI of the transceiver i and V_i denotes the expiration date of the cryptography material provided by the CA. Note that the equations 2 and 4 are modified accordingly, replacing I_i with l_i.

C. Online Phase

The *Online Phase* of *SecureAIS* is executed at run-time when two AIS transceivers require the exchange of secure binary messages. The protocol has been designed as an application that runs over AIS, through the exchange of dedicated binary messages.

The key agreement protocol has been thought of as the combination of well-known schemes, whose security has been already proved in the past. Specifically, it adopts a certified ECDH scheme, where the ECQV scheme is used as the certification mechanism. The logical sequence diagram of the protocol is depicted in the following Figure 4.

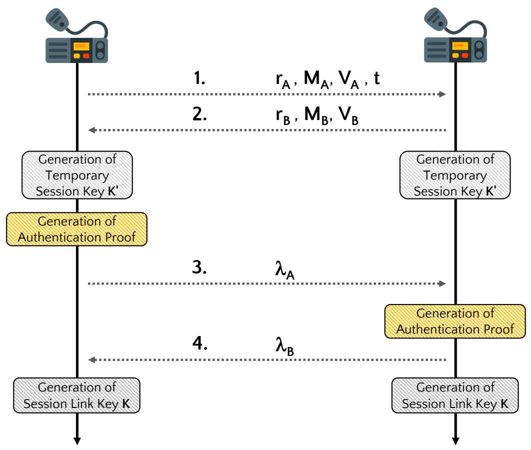

1.
r_A, M_A, V_A, t
2.
r_B, M_B, V_B
Generation of Temporary Session Key K'
Generation of Temporary Session Key K'
Generation of Authentication Proof
3.
λ_A
Generation of Authentication Proof
4.
λ_B
Generation of Session Link Key K
Generation of Session Link Key K

Figure 4. Sequence Diagram of the Online Phase of the *SecureAIS* scheme.

Overall, the online phase of *SecureAIS* requires two stages. In the first stage, the legitimate entities exchange the cryptography materials require to obtain the public key of the remote entity and compute the shared key, according to the ECDH scheme. Following, the second stage is meant to provide mutual authentication and agree on a symmetric *session key*, to be used to secure the following binary messages.

The detail of the messages exchanged between the transceivers and the operations required to complete the protocol are detailed below.

We assume that the AIS Transceiver A would like to establish a secure key with the AIS Transceiver B, to secure all the following binary messages to be exchanged. To this aim, the entity A triggers the *SecureAIS* protocol by transmitting a unicast binary message to the entity B. This message includes a randomly generated nonce, namely r_A, the implicit certificate M_A received from the Maritime Authority during the Setup Phase, and the correspondent implicit certificate validity time V_A. The message also includes a *Security Level Indicator* t, specifying the cryptographic security level desired by the entity A.

On the reception of the triggering message, the entity B first verifies the consistency of the information provided by the remote entity A. First, it extracts the identity I_A reported in the MMSI Source ID of the AIS message. Then, it checks that the expiration date of the cryptography materials is posterior to the actual date. Moreover, the entity B verifies that the *Security Level Indicator* indicated by A is equal or higher than the minimum security level locally supported. If one of the conditions reported above is not fulfilled, a *failure* message is delivered to the entity A, indicating the reason for the rejection.

If the above checks are successfully passed, the entity B stores the nonce r_A locally for future use. Then, it triggers the ECQV scheme to generate the public key of the remote entity A. To this aim, it executes the operation reported in the following Eq. 8:

$$P_A = C + H(M_A jj l_A) \quad M_A. \tag{8}$$

As demonstrated by the authors in the reference papers [11], [16], the contribution of the CA through its public key C in the computation of the public key of the remote entity is enough to ensure the authenticity of the key P_A locally generated.

Then, B executes the ECDH scheme to generate a *temporary session key*, as depicted in the following Eq. 9:

$$K_A^0 = K^0 = KDF(p_B \quad P_A), \tag{9}$$

where KDF refers to a generic *Key Derivation Function*, used to generate a fixed-length key from a generic input string.

Now, B delivers to A a message containing its implicit certificate M_B, the related expiration date v_B, and a randomly generated nonce r_B.

On the reception of the message, similar to the operations executed by B, the entity A first verifies the consistency of the information provided by the remote entity B. Specifically, it checks that the expiration date of the cryptography materials reported in the *identification string* is posterior to the actual date. If one of the conditions reported above is not fulfilled, a *failure* message is delivered to the entity B, indicating the reason for the rejection.

If the above checks are successfully passed, the entity A stores the nonce r_B locally for future use. Then, it generates the public key of the remote entity B, by executing the operation reported in the following Eq. 10:

$$P_B = C + H(M_B jj l_B) \quad M_B. \tag{10}$$

Then, A executes the ECDH scheme to generate a *temporary session key*, as depicted in the following Eq. 11.

$$K_B^0 = K^0 = KDF(p_A \quad P_B). \tag{11}$$

A further proceeds to generate the cryptography values used to authenticate the message exchange. To this aim, it computes the authentication proof λ_A, according to the following Equation 12:

$$\lambda_A = HMAC(r_B jj M_B jj P_B jj r_A jj M_A jj P_A, K_A^0), \tag{12}$$

where the terminology $HMAC(m, K)$ refers to the use of a generic HMAC function on the message m using the key K.

The authentication proof λ_A is then delivered to B in a dedicated binary message.

On the reception of the second message from the entity A, the AIS transceiver B authenticates the entity A by verifying the condition in the following Eq. 13

$$HMAC\left(r_B || M_B || P_B || r_A || M_A || P_A, K_B^0\right) == \lambda_A, \tag{13}$$

If this check is verified, B can be sure that A has the private key corresponding to the delivered public key, and that it shares with B a secure symmetric key.

B generates its own authentication materials, namely λ_B, according to the following Eq. 14.

$$\lambda_B = HMAC(r_A || M_A || P_A || r_B || M_B || P_B, K_B^0), \tag{14}$$

Then, it delivers λ_B to A in a dedicated binary message.

Finally, B can generate the final *session key* K, according to the following Eq. 15.

$$K_B = K = KDF\left(K_B^0 || r_A || r_B\right), \tag{15}$$

where r_A and r_B are the two nonces exchanged in the first message.

At the reception of this last message from B, the AIS transceiver A can establish the authenticity of the remote entity, by verifying the following Eq. 16.

$$HMAC\left(r_A || M_A || P_A || r_B || M_B || P_B, K_A^0\right) == \lambda_B, \tag{16}$$

If this check is verified, A can be sure that B is authentic, and that it shares with A a secure symmetric key.

Finally, A generates the final *session key* K, according to the following Eq. 17.

$$K_A = K = KDF\left(K_A^0 || r_A || r_B\right), \tag{17}$$

where r_A and r_B are the two nonces previously exchanged. According to the cryptographic principles of the ECDH algorithm, $K_A = K_B = K$ is the final shared session key that will be used to secure any binary message exchanged between the two AIS transceivers.

We notice that the above-described *SecureAIS* protocol should be executed by a specific couple of AIS transceivers only at the first interactions between the two devices. For subsequent messages exchange, if the certificates are still valid, the two devices can re-use the same key previously established. Only when the certificates expire and are replaced with new cryptography materials, one of the AIS transceivers should trigger a new instance of the *SecureAIS* scheme.

Finally, we highlight that all the secure AIS messages, secured through the key negotiated via *SecureAIS*, can be delivered as standard binary messages (Message Type 6) or using a dedicated *Application Identifier*, chosen among the ones actually unused by the standard.

V. Security Analysis

The security properties achieved by *SecureAIS* have been verified via the automated tool ProVerif [17]. We recall that ProVerif is widely adopted in the literature to formally verify the security properties achieved by cryptographic protocols, assuming the same Dolev-Yao attacker model assumed in this paper (see, for instance, [18]).

SecureAIS has been implemented in the ProVerif tool, to verify two main properties: (i) the secrecy of the negotiated key, and (ii) the mutual authentication between the AIS transceivers. According to the logic of the ProVerif tool, four main events have been defined.

1) *begin_SecureAIS_A*: Indicating that AIS transceiver B believes it has initialized a protocol instance with AIS transceiver A.
2) *begin_SecureAIS_B*: Indicating that AIS transceiver A believes it has initialized a protocol instance with AIS transceiver B.
3) *end_SecureAIS_A*: Indicating that AIS transceiver B believes it has finalized the protocol with AIS transceiver A.
4) *end_SecureAIS_B*: Indicating that AIS transceiver A believes it has finalized the protocol with AIS transceiver B.

The following output messages are provided by ProVerif to identify the fulfillment of security properties:

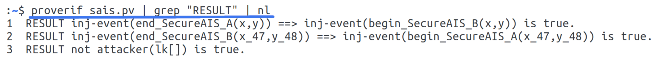
inj-event(last_event ()) ==> inj-event(previous_event ()) is true: Meaning that the function *last_event* is executed only when another function, namely *previous_event* is really executed;

inj-event(last_event ()) ==> inj-event(previous_event ()) is false: Meaning that when the function *last_event* is executed, it is not always true that the function *previous_event* has been executed before.

not attacker(elem[]): Meaning that the attacker is not in possession of the value of *elem*;

attacker(elem[]): Meaning that the attacker is in possession of the value of *elem*;

Figure 5 reports the output of the ProVerif tests.



```
:~$ proverif sais.pv | grep "RESULT" | nl
1  RESULT inj-event(end_SecureAIS_A(x,y)) ==> inj-event(begin_SecureAIS_B(x,y)) is true.
2  RESULT inj-event(end_SecureAIS_B(x_47,y_48)) ==> inj-event(begin_SecureAIS_A(x_47,y_48)) is true.
3  RESULT not attacker(lk[]) is true.
```

Figure 5. Verification of security properties of *SecureAIS* using ProVerif.

Note that the AIS transceiver B completes successfully the protocol only when it is the AIS transceiver A that initiated it. Indeed, if AIS transceiver A believes it has terminated the protocol with AIS transceiver B, this latter is the correct entity executing the protocol, and vice-versa. Thus, the two devices A and B are mutually authenticated. We also notice that the negotiated key, namely K, is verified as *secret*. Overall, the positive outcomes provided by the queries allowed us to verify that *SecureAIS* is secure against Man-In-The-Middle and replay attacks and that the negotiated key is secret. The source code of the formal verification of *SecureAIS* has been also released as open-source [9], to allow interested readers to verify our claims.

VI. Implementation and Performance Evaluation

To practically show the feasibility of the proposed scheme for real devices using AIS, we implemented a real Proof-of-Concept of *SecureAIS* in the GNURadio ecosystem. Our proof-of-concept has been realized using the Ettus Research X310 SDR, featuring a UBX160 daughterboard [19]. Such a setup is a common and reasonable choice when implementing and testing protocols for communication technologies such as AIS, requiring hardware working on dedicated frequency bands [20].

Our implementation leverages and extends the AIS transmission module provided by the authors in [4], with an application that dynamically executes the security operations required by *SecureAIS*, encapsulates the information in standard-compliant AIS messages, generates the bit-sequence, and instructs the radio to transmit the messages according to the AIS standard specifications. Note that, for the implementation of *SecureAIS*, we used SHA256 and HMAC-SHA256 functions and the hash and the HMAC operations, and the KDF2 key derivation function. On the reception side, we used the *gr-ais* module available at [21], modified in a way to synchronize with the transmitting module and achieve 100% reliability in the radio operations.

We remind that also the source code of the modified transmitter and receiver in the GNURadio development toolkit have been released as open-source at the link [9]. This enables researchers and maritime companies to reproduce our results, use the module on top of existing AIS transceivers, and further develop secure AIS applications.

We performed several experimental tests to evaluate the feasibility and overhead of *SecureAIS* for devices using AIS. All the tests reported in this section have been performed using wireless RF operations. Further, in line with the TDMA method discussed in Section II, we used continuous AIS data streams for a maximum of 3 slots (compatible with Class-B devices), triggering a new RF transmission when more data need to be delivered. First, we investigated the time required to execute all the operations required by *SecureAIS*, assuming to work with the maximum available security level, i.e., 256 bits (worst-case). For this test, we measured for 20 times the time required to complete each of the steps included in *SecureAIS*, being these either processing tasks or communication tasks. The results are illustrated through the following Figure 6. We notice that the most time-consuming operations are the RF ones, accounting for over the 99% of the total overhead ($3,133$ ms over $3,164$ total). This is due to the architecture of our proof-of-concept, where the bit-stream generated on the laptop is delivered to the SDR via an Ethernet connection, to accomplish filtering, modulation, and transmission, and then, similar but reverse operations are executed on the reception. In real AIS transceivers, such delays are not involved, and this definitively reduces the cost of the protocol. We also notice that ECC operations are characterized by a reduced cost, accounting for only 0.031 s with the highest security level. We also investigated the time required to complete *SecureAIS*, with different security levels. Each security level has been tested 20 times, and the values are reported using the 95% confidence intervals. We notice that the overhead due to the increase in the security level is mainly due to the additional AIS time-slots occupied by the transmitting AIS transceiver, and not to the computational burden posed by the security operations. Indeed, while the lowest security level (80 bits) requires 10 AIS time-slots (5 per each transceiver), the security level 128 requires 14 time-slots, the security level 192 requires 16 time-slots, and, finally, as shown in the previous test, the security level 256 requires 20 time-slots (10 for each entity). Note that the slighter difference existing between the security level 128 and 192 is due to the reduced amount of slots necessary to switch from the two levels. We remark that a consistent part of the overhead is due to the network architecture used for the Proof-of-Concept, while using real AIS transceiver these delays could be significantly reduced.

Finally, we compared *SecureAIS* with a scheme based on the same logic, but using X.509 explicit certificates, looking at the number of AIS time-slots required to complete the key agreement scheme. The results are reported in the following Figure 8. We notice that the proposed configuration of *SecureAIS* is the one that requires the least amount of time-slots. At the same time, including X.509 certificates would require an overwhelming number of time-slots, leading to consistent time and bandwidth consumption. Specifically, assuming the higher security level of 256 bits, the delivery of X.509 requires 96 time-slots. Instead, the usage of ECQV implicit certificates allows reducing the number of messages to only 20 time-slots, achieving a more effective bandwidth usage and reduced time to have a shared session key.

VII. Conclusion and Future Work

In this paper, we presented *SecureAIS*, a standard-compliant, software-only, key establishment protocol enabling two AIS transceivers to agree on a pairwise session key, to be used by upper-layer applications for security purposes (confidentiality and/or implicit key authentication). *SecureAIS* integrates existing building blocks, such as ECDH and ECQV techniques, and its security has been formally proven via the automated tool ProVerif. Our proof-of-concept realized using GNURadio and Ettus Research X310 SDR demonstrates that, compared to existing solutions, *SecureAIS* requires the least amount of slots, i.e., only 20 AIS time-slots when used to obtain a shared secret key characterized by the highest security level, 256 bits—incurring in less than 20% of the overhead required by traditional X.509 certificates.

Future work include the integration of *SecureAIS* in hand-held AIS transceivers, characterized by limited computational capabilities, the management of certificate revocation issues, and a performance evaluation in harsh link conditions.

Acknowledgements

This publication was partially supported by awards NPRP-S-11-0109-180242 and NPRP X-063-1-014 from the QNRF-Qatar National Research Fund, a member of The Qatar

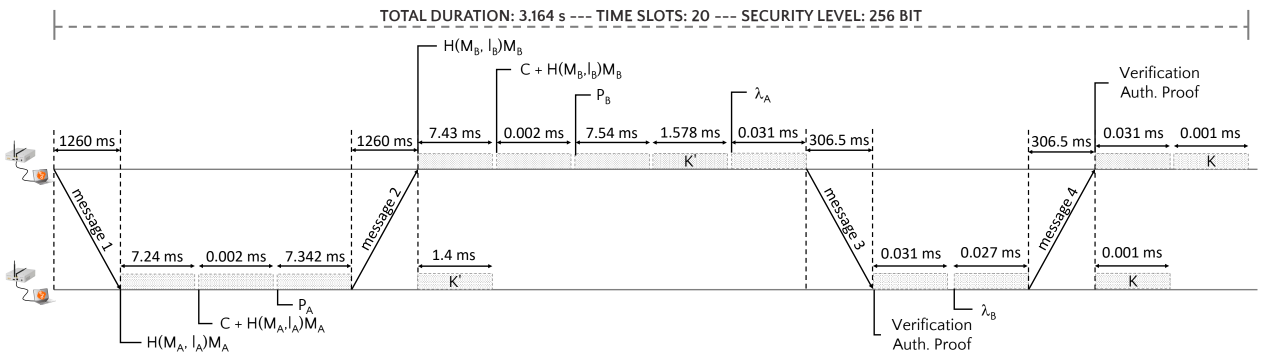


Figure 6. Time required to complete each step in *SecureAIS*, assuming the maximum security level of 256 bits—plots are not in scale.

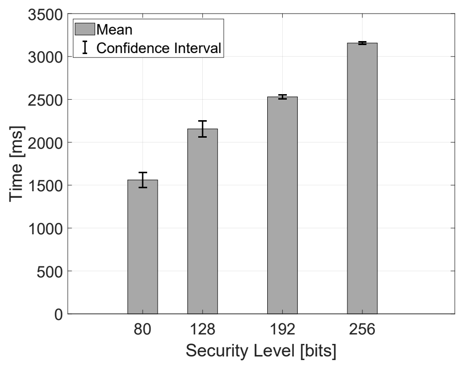


Figure 7. Time to complete *SecureAIS*, with different security levels.

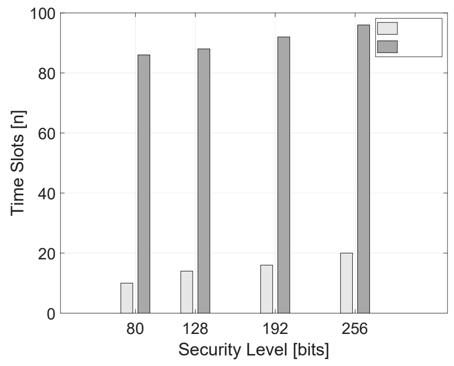


Figure 8. Comparison between *SecureAIS* using ECQV and X.509 certificates, in terms of required time-slots.

Foundation. The information and views set out in this publication are those of the authors and do not necessarily reflect the official opinion of the QNRF.

References

[1] R. L. Shelmerdine, "Teasing out the detail: How our understanding of marine AIS data can better inform industries, developments, and planning," *Marine Policy*, vol. 54, pp. 17 – 25, 2015.

[2] M. Caprolu, R. Di Pietro, S. Raponi, S. Sciancalepore, and P. Tedeschi, "Vessels Cybersecurity: Issues, Challenges, and the Road Ahead," *IEEE Communications Magazine*, 2020.

[3] Marine Traffic. (2020) Marine Traffic: Global Ship Tracking Intelligence. [Online]. Available: https://www.marinetraffic.com/en/ais/home

[4] M. Balduzzi, A. Pasta, and K. Wilhoit, "A Security Evaluation of AIS Automated Identification System," in *Proc. of the Annual Computer Security Applications Conference (ACSAC)*, 2014, p. 436–445.

[5] J. Hall, J. Lee, J. Benin, C. Armstrong, and H. Owen, "IEEE 1609 Influenced Automatic Identification System (AIS)," in *IEEE Vehicular Technology Conf.*, May 2015, pp. 1–5.

[6] R. Pelich, M. Chini, R. Hostache, et al., "Large-Scale Automatic Vessel Monitoring Based on Dual-Polarization Sentinel-1 and AIS Data," *Remote Sensing*, vol. 11, no. 9, p. 1078, 2019.

[7] Shwu-Jing Chang, "Vessel identification and monitoring systems for maritime security," in *IEEE Int. Carnahan Conference on Security Technology*, Oct 2003, pp. 66–70.

[8] A. Goudossis and S. K. Katsikas, "Towards a secure automatic identification system (AIS)," *Journal of Marine Science and Technology*, vol. 24, no. 2, pp. 410–423, Jun 2019.

[9] Cybersecurity Research and Innovation Lab (CRI-LAB), "Open-source code of the implementation of SecureAIS in the GNURadio OS and code of the ProVerif tool," https://github.com/cri-lab-hbku/secureais, 2020, (Accessed: 2020-02-08).

[10] International Telecommunications Union (ITU), "Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile frequency band," ITU, ITU-R Reccomendation, 2014.

[11] D. Brown, R. Gallant, and S. A. Vanstone, "Provably Secure Implicit Certificate Schemes," in *Financial Cryptography*, P. Syverson, Ed. Springer Berlin Heidelberg, 2002, pp. 156–165.

[12] S. Sciancalepore, G. Piro, G. Boggia, and G. Bianchi, "Public Key Authentication and Key Agreement in IoT Devices With Minimal Airtime Consumption," *IEEE Embedded Systems Letters*, vol. 9, no. 1, pp. 1–4, Mar. 2017.

[13] G. Oligeri, S. Sciancalepore, O. A. Ibrahim, and R. Di Pietro, "Drive Me Not: GPS Spoofing Detection via Cellular Network: (Architectures, Models, and Experiments)," in *Proc. of Conf. on Security and Privacy in Wireless and Mobile Netw.*, 2019, p. 12–22.

[14] K. Jansen, M. Schäfer, D. Moser, V. Lenders, C. Pöpper, and J. Schmitt, "Crowd-GPS-Sec: Leveraging Crowdsourcing to Detect and Localize GPS Spoofing Attacks," in *2018 IEEE Symposium on Security and Privacy (SP)*, May 2018, pp. 1018–1031.

[15] M. Rocchetto and N. O. Tippenhauer, "CPDY: extending the Dolev-Yao attacker with physical-layer interactions," in *International Conference on Formal Engineering Methods*. Springer, 2016, pp. 175–192.

[16] D. Brown, M. J. Campagna, and S. A. Vanstone, "Security of ECQV-Certified ECDSA Against Passive Adversaries," Cryptology ePrint Archive, Report 2009/620, 2009, https://eprint.iacr.org/2009/620.

[17] B. Blanchet, "Automatic Verification of Correspondences for Security Protocols," *Journal of Computer Security*, vol. 17, no. 4, pp. 363–434, 2009.

[18] P. Tedeschi, S. Sciancalepore, A. Eliyan, and R. Di Pietro, "LiKe: Lightweight Certificateless Key Agreement for Secure IoT Communications," *IEEE Internet of Things J.*, vol. 7, no. 1, pp. 621–638, 2020.

[19] Ettus Research, "UBX160 Daughterboard," https://www.ettus.com/product/details/UBX160, 2020, (Accessed: 2020-02-08).

[20] 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*, pp. 1–1, 2019.

[21] GNURadio, "AIS Receiver with GNURadio Common Blocks," https://github.com/juan0fran/ais_rx, 2020, (Accessed: 2020-02-08).