Authenticated Encryption for Janus-Based Acoustic Underwater Communication

Branislav Petrović
Norwegian University of Science and Technology (NTNU)
P.O.Box 8900, Torgarden, 7491 Trondheim, Norway
branislp@stud.ntnu.no

Bálint Zoltán Téglásy
Norwegian University of Science and Technology (NTNU)
P.O.Box 8900, Torgarden, 7491 Trondheim, Norway
balint.teglasy@ntnu.no

Sokratis Katsikas
Norwegian University of Science and Technology (NTNU)
P.O.Box 191, 2802 Gjøvik, Norway
sokratis.katsikas@ntnu.no

***Resumen*—Wireless underwater communications commonly utilize acoustic waves due to their superior robustness and range compared to radio waves. But usage of acoustic waves introduces some constraints, such as a low data rate and high packet loss. In addition, most information sent using acoustic waves under water today is unencrypted and unauthenticated. With the development of underwater environments, there is an increased need for data protection among marine operators, since the underwater threat landscape is rapidly broadening with new kinds of attacks, such as eavesdropping, routing attacks, and data tampering. To overcome these problems, in this paper, we propose two security schemes integrated with the first standard for acoustic underwater communication, Janus. The aim is to counteract the threats, bearing in mind the limitations of the acoustic communication channels. The proposed schemes are based on symmetric cryptography. The ultimate goal is to provide a way for underwater nodes to exchange authenticated and encrypted information.**

***Index Terms*—underwater acoustic communication, authenticated encryption, Janus**

I. Introduction

Wireless underwater communication networks are rapidly increasing in quantity and there exist several types of underwater networks that serve different purposes. Currently, few standardized solutions for underwater communication exist and not all are available for public use. The currently most established digital underwater communication standard is Janus [5], a physical-layer acoustic standard that is robust and makes interoperability among maritime operators possible. However, Janus by itself provides no security mechanism, neither in the form of encryption or authentication. As the Janus standard already serves as a basis for wireless communication, in this paper we focus on adding security mechanisms to it. Authentication of messages and entities is an important requirement, in addition to encryption, since not only confidentiality, but also integrity of data is to be ensured. Authenticated encryption is a mechanism that provides both integrity and confidentiality.

The threats to underwater acoustic communication are similar to those faced by radio communication above the sea surface. Similar techniques are used to disrupt the propagation of the waves and, thereby, also the communication. In addition, routing protocols that forward packets based on channel quality and the possibility to reach adjacent nodes, are also in use in underwater networks. Protocols of this kind can be manipulated by malicious users to cause disruptions in networks.

Ghannadrezaii et al. [4] classify threats to underwater communication into three main categories: Eavesdropping, Data tampering, and Routing attacks, while Domingo [3] also considers Jamming, Wormhole, and Sybil attacks.

The main constraints for providing security features under water are low data rate and high packet loss in this environment. These constraints make the use of many standardized encryption and authentication schemes infeasible due to their complexity and scale. Consequently, the authenticated encryption mechanisms must be computationally inexpensive and reduce the overhead to a minimum.

Bearing in mind the threat landscape and the constraints of the communication under water, it is realistic to assume that the capabilities of eventual adversaries enable performing the attacks mentioned above. Therefore, incorporating security mechanisms in underwater communication standards is necessary.

In this paper, we propose two solutions for the authenticated encryption in Janus-based underwater communications. The first proposal is based on the well-known scheme CCM - Counter and CBC MAC (CBC - Cipher Block Chaining, MAC - Message Authentication Code) [13], and the second proposal is based on a more recent scheme AEGIS [1]. We define Janus-compatible protocols capable of realizing these two authenticated encryption schemes and give arguments in favor of their usage.

The structure of the paper is the following: Section II discusses the Janus standard and the previous proposals that introduce some security mechanisms in it. In Section III, we propose two authenticated encryption schemes that could be used in Janus and give the arguments in favor of using them. Section IV concludes the paper.

II. Background and related work

The Janus standard allows for a bandwidth of 80 bps and a range of 10 km. The baseline packet is 64 bits long. 34 of these bits are reserved for user data, and are called the Application Data Block (ADB). The rest of the packet consists of communication overhead that specifies different communication properties. The very limited amount of user data that can be transmitted in a packet makes Janus best suited for small data exchanges, such as Command-and-Control or status messages. If larger amounts of data are to be sent, it must be either distributed into several packets or be sent as a *cargo* immediately following a packet. If several packets are used, the additional overhead must be encoded and modulated for each packet, resulting in additional use of computing resources and increased latency. If a cargo is specified, the channel is reserved during the transmission of the cargo and no other communication is to happen on the network while it is being transmitted.

Most current research of wireless underwater security considers encryption and authentication schemes based on symmetric cryptography and pre-shared information, as this is the only model that is realizable with today's physical underwater infrastructure. Nevertheless, methods based on public key cryptography that would be applicable in a hypothetical underwater PKI (Public Key Infrastructure) have also been proposed (see, for example, [4], [8]).

In order to provide a certain level of security, the recently published subclass of Janus, Venilia [11], specifies an encryption scheme for Janus packets using the custom-made Tiny Underwater Block (TUB) cipher [10]. With Venilia, 27 of the 34 bits in the Janus ADB are encrypted (27 bits is the block size of the TUB cipher). These 27 bits consist of an 8-bit message (so-called pre-canned message), a source address and destination address of 7 bits each, and an additional 5-bit CRC (Cyclic Redundancy Check). The remaining 7 bits in the ADB house a 5-bit IV (Initialization Vector) and a 2-bit epoch identifier, both of which are used as input to the TUB cipher. The 8-bit message field allows for 256 unique messages to be processed. These messages must be stored in a pre-shared code book that is put on all devices in a network.

A drawback of Venilia is that its operation is restricted to 8-bit Command-and-Control and status messages, which are stored in a predefined code book at each device in a communication network. The fact that Venilia does not define the use of cargo packets further restricts its communication capabilities. Consequently, if 30-bit Maritime Mobile Service Identity (MMSI) is going to be used for entity authentication, then Venilia cannot be used for encryption.

With Venilia, it is assumed that authentication has already taken place before the use of Venilia. Consequently, many kinds of attacks can be launched against communication that does not employ authentication, of which the most prominent ones are MITM (Man-In-The-Middle)-based attacks.

The fact that the IV is only used for key derivation and not in the encryption process, implies that the TUB cipher is used in the ECB (Electronic Codebook) mode, which is vulnerable to statistical attacks, although it is worth noting that the key management of Venilia provides certain protection against such attacks.

An approach using public key cryptography is described in [4]. Here, key establishment between nodes in an underwater network is performed using the Elliptic Curve Diffie-Hellman key exchange protocol. After two nodes have established an identical key each, they then encrypt subsequent communications with a symmetric encryption algorithm, such as AES (Advanced Encryption Standard). This scheme permits secure underwater machine-to-machine communication between the acoustic nodes in underwater networks. However, it assumes the existence of an underwater PKI, which is not specified.

The first complete proposal of an authentication procedure based on Janus is given by Téglásy et al. [9]. Here, two devices, initially unknown to each other, first identify each other as friend or foe by determining whether they possess the same pre-shared key. They achieve this by exchanging a timestamp, a clock accuracy descriptor, and two SYN and ACK flags in the ADB of the Janus packet. These values are encrypted with the 32-bit block version of the RC5 cipher using a pre-shared long-term key K_n of at least 128 bits in length. The authentication is based on the validity of the timestamps exchanged by the devices and the fact that the sending device would be unable to encrypt the message without K_n. Assuming that the devices' clocks were synchronized during the exchange of K_n at the start of the mission, a device checks if the timestamp it received is within the expected bounds compared to the mission duration. The expected bounds are adjusted according to possible deviations in clock synchronisation between the devices and the expected maximum distance that a message can travel. If the timestamp is valid, the receiver sends its own timestamp and clock accuracy descriptor back to the originating device, encrypted with the same key K_n. The protocol is shown in Fig. 1.

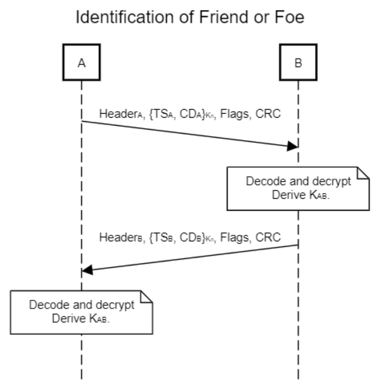
Identification of Friend or Foe
A
B
Header$_A$, {TS$_A$, CD$_A$}$_{K_n}$, Flags, CRC
Decode and decrypt Derive K$_{AB}$.
Header$_B$, {TS$_B$, CD$_B$}$_{K_n}$, Flags, CRC
Decode and decrypt Derive K$_{AB}$.

Figura 1. Protocol for identification of friend or foe [9].

To protect the communication in the case of compromise of K_n, the exchanged values are later used to compute a session key K_{AB} with a custom Key Derivation Function (KDF). The KDF also utilizes RC5, albeit with a block size of 128 bits, to produce a 256-bit ciphertext. K_{AB} allows communication to remain secure if K_n is compromised after the establishment of K_{AB} since subsequent messages will be encrypted with K_{AB} instead of K_n. It is also infeasible for attackers to obtain previous session keys if they obtain K_n and they have not eavesdropped on previous runs of the protocol. Since K_{AB} is derived from a timestamp, a new timestamp sent by an attacker to a legitimate node as an authentication attempt will result in a different key due to the passing of time and, thus, a different timestamp.

As one device may derive session keys with many other devices, it is necessary to create a mapping of the derived session keys to their corresponding device identities. This is achieved with a shared lookup table at each device, containing all other devices' identities with their corresponding long-term keys. When a device receives a message that is encrypted with a certain long-term key, it will attempt to decrypt the message with the long-term keys in its lookup table, sequentially. If one of the keys yields a successful decryption, the key that yielded the decryption is used with the KDF to derive a session key, and the session key is stored along with the identity and the long-term key in the table. The identity has the form of MMSI, which is a standard format used for tracking ships above the sea surface. The length of the MMSI is 30 bits.

Although this protocol provides authentication, encryption, and key establishment, it has certain limitations. They are outlined below.

Timestamp forgery: Authentication of messages that relies solely of the validity of timestamps is vulnerable since timestamps can be forged by an attacker relatively simply. As long as the attackers used a valid K_n to encrypt their message, they will be able to successfully establish a session key with a valid device. Moreover, if attackers obtain K_n before the protocol is run, they can decrypt any message that is encrypted with K_n, as well as derive K_{AB}.

Timestamp replay: Another concern is replay of timestamps. An attacker can relatively simply forge the correct timestamp by observing when the messages in the legitimate protocol are transmitted and recording the time at that moment. Since the acceptance of timestamps at legitimate devices is adjusted to allow errors caused by currents, clock drift, encryption, and decryption delays, there is a possibility that the forged timestamp will be accepted as legitimate.

Limited forward secrecy: After the establishment of K_{AB}, messages encrypted under it remain secure even if attackers possess K_n. However, this is only true if the attackers have not eavesdropped on or intercepted any previous messages. If previous protocol runs with messages encrypted with K_n are eavesdropped and stored, compromise of K_n at any point in time will give attackers the possibility to decrypt the messages and calculate previous session keys that depend on K_n.

Limited encryption strength: RC5 was chosen because of its security provided by a long key and its relatively simple software implementation. However, Biryukov et al. [2] show that, with partial differential cryptanalysis, it is possible to derive all 25 subkeys for the 12 rounds of the 32-bit block version of RC5, with 2^{44} chosen plaintexts. Even though it is not likely that this amount of messages will be encrypted under the same key, employing a stronger encryption algorithm would also eliminate this theoretical possibility.

Poor scalability: The lookup table stored on each device requires substantial storage space and memory to store, read, and write to. Additionally, attempting to decrypt a received message with all long-term keys consumes large amounts of time and computational resources, resulting in poor performance, especially in resource-constrained devices, such as sensors. If large networks with many devices are to be supported by an authentication protocol of this kind, the scalability and performance must be improved.

Some limitations mentioned above, such as limited encryption strength and poor scalability, have been addressed in our proposals. Further study is needed in order to conclude whether our proposed solutions also address the timestamp replay/forgery and forward secrecy challenges in a satisfactory way.

III. THE NEW AUTHENTICATED ENCRYPTION SCHEMES FOR JANUS

III-A. Authenticated Encryption with CCM

Given the security and bandwidth requirements, a potentially suitable algorithm for providing assurance of the confidentiality and the integrity of Janus data is the CCM mode of operation for block ciphers, proposed by Whiting et al. [13] and standardized as Special Publication 38-C by NIST [7]. The counter mode it applies for encryption allows the avoidance of transmitting complete blocks of the underlying block cipher, even though the data must be padded for partitioning into complete blocks. The only additional overhead in the data that is sent are the nonce and the MAC tag. It is therefore a good option for our needs as it is a standard that provides a high level of security, and the Janus baseline packet and cargo can be formatted as input to the CBC-MAC and encryption algorithms.

According to [7], CCM is only defined for block ciphers with a block size of 128 bits. Currently, the only approved cipher for the algorithm is AES. There are three inputs to the algorithm: the plaintext P (also called the payload) to be authenticated and encrypted, the associated data AD that will only be authenticated and not encrypted, and a nonce N, which is a unique value associated with the other two inputs. Typically, the payload consists of user data that needs to be confidential, while the associated data are packet headers that remain unencrypted for the proper functioning of routing mechanisms in networks. In our case, the payload consists of the ADB in the Janus baseline packet and optional cargo, while the associated data is the 22-bit preamble before the ADB, and the 8-bit checksum. The nonce can be the timestamp of an autonomous underwater vessel or a random number. CCM defines two major operations: generation-encryption, in which the MAC tag is generated and the payload is encrypted, and decryption-verification, where the payload is decrypted and the MAC tag is verified.

We consider the example where only the 64-bit baseline Janus packet is to be processed by CCM. The length of the secret key K_n is 256 bits. The MAC tag T should, by the specification, have a length of at least 4 bytes to prevent forgery attacks. The octet length of N, n, should, by the specification, be at least 7. The total length of AD is 30 bits, while the length of P is 34 bits.

III-A1. Argumentation for the use of CCM: CCM is used first and foremost due to its familiarity as a mode of operation for AEAD (Authenticated Encryption with Associated Data) and its long-lasting and widespread use in WLAN (Wireless Local Area Networks). Despite the age of the algorithm, very few practical attacks against both confidentiality and authenticity have been developed, meaning that it still provides a high level of security. Another argument for its usage in underwater networks is its ability to partially encrypt a message, while authenticating both the encrypted and unencrypted parts. Since the Janus packets in our authentication protocol have messages of this form (i.e., it is possible to format the Janus header as associated data, while only encrypting the ADB), formatting our messages for the structure of CCM is relatively simple.

Jonsson [6] provides a formal analysis of CCM with the conclusion that a high level of confidentiality and authenticity is provided in line with other standardized modes of operation for authenticated encryption, such as GCM (Galois/Counter Mode) or OCB (Offset Codebook Mode). The attractive properties of CCM are listed below:

1. A specific mechanism that handles parts of messages that are only to be authenticated and not encrypted is provided by default. This is done without additional ciphertext overhead, something that requires enhancements in many other authenticated encryption modes.
2. AES is used only in the forward direction for encryption and its inverse is never used. This contributes to the reduction of the code size of implementation.
3. The `ctr` and `cbcmac` algorithms are widely deployed and have been in use for a long time, meaning that CCM is based on well known and proved technology. Existing implementations are also highly optimized.
4. All intellectual property rights have been released to the public domain, making CCM freely available for any purpose.

Regarding authenticity, Jonsson claims that it is hard to extract any non-trivial information about the input blocks and the output blocks of the CBCMAC algorithm, even if all the plaintexts of a message exchange are known. With respect to confidentiality, the goal of an attacker would again be to distinguish a ciphertext from a bit string chosen uniformly at random from the set of all possible bit strings of the given length. The two ways that this can be done are either that the attacker executes a birthday attack against the `ctr` output blocks or that an anomaly occurs within the `cbcmac` computation, such as an internal collision or a MAC tag that is identical to some `ctr` output block.

III-A2. Application in Janus-based communication: Regarding the application in Janus, CCM can be directly applied, using the smallest recommended values of t, n, a, and p, such that the standardized levels of confidentiality and authenticity are provided. However, the output of the processing of a Janus baseline packet cannot itself fit into a single baseline packet due to the need to transmit the nonce N and the MAC tag T in addition to the associated data AD and the encrypted payload C. Instead, either a cargo must be specified or several baseline packets must be used. Which option is better depends on several factors, such as the scale of the network and the amount of data that the devices send on average. Since the aim of Janus is to be an open standard, we believe that users with different requirements may wish to use either option. Following are descriptions of the incorporation of CCM into the protocol for authentication of underwater assets, using both options - without cargo and with cargo.

III-A3. Without usage of cargo: To perform full authentication i.e., both authentication of messages and entity authentication, it is necessary to transmit a 29-bit timestamp TS, a 3-bit clock accuracy descriptor CD, and two 1-bit SYN and ACK flags F, as well as the MMSI of both device A and B, in each direction. The transmission of both MMSIs allows the sending device to authenticate itself and to indicate which device the message is designated for. Thus, there is no longer a need to store session keys at each device. This improves scalability, as the storage of the lookup tables and the lookup procedures would require much storage space and processing as the network grows. For CCM, the required input at the receiving device are the nonce N, the associated data AD, and the ciphertext C, which consists of the plaintext P and MAC tag T, both encrypted. Hence, our goal is to partition the values used in the authentication protocol into the format of CCM.

For CCM to function, N must have a length of at least 56 bits and cannot fit into the 34-bit ADB. However, a way to resolve this is to set an initial length of N to 32 bits and then duplicate that value locally at each device to produce a 64-bit string, which is a valid length for the generation-encryption and decryption-verification operations. The entropy of N will in this case still be the same as for a 32-bit string, but this is satisfactory for most applications.

Regarding the associated data, F should be authenticated, as it provides relevant status information in the protocol. Taking this into account, AD consists of the 22-bit Janus header and the two 1-bit flags, so it is 24 bits long. In our case, the Janus header will remain the same for all CCM-related packets sent in one direction that are part of the authentication protocol. This allows for the usage of the Janus header of any packet to directly be included in AD, without the need to partition AD in the ADB. Thus, the first baseline packet can be used to transport both N and the 22 bits of AD that are made up of the Janus header. The two remaining bits of AD can either be transmitted in the same packet or in the next one.

The timestamp and clock accuracy descriptor consist of 32 bits in total and can therefore fit in the ADB together with the flags F. TS and CD can also directly be encrypted in the counter mode of CCM such that a 32-bit ciphertext is produced. Hence, the second packet in the protocol is used to transport TS and CD in encrypted form and F in plaintext.

The MMSIs have a length of 30 bits each and must therefore be located in their respective baseline packets. Therefore, the third and fourth packets are used to transport MMSI$_A$ and MMSI$_B$.

The authentication tag T is the final element that needs to be transmitted. It is possible to set the size of T to 4 bytes and conveniently place it in the ADB.

The complete protocol can be seen in Fig. 2. As each participant transmits five 64-bit baseline packets, the theoretical delay will be $64 \cdot 5/80 = 5s$, in addition to the propagation delay of the acoustic signals, which depends on the distance between the participants.

III-A4. With usage of cargo: The usage of a cargo would allow for transmitting all the required data in a single packet, thus avoiding the need to transmit AD and a CRC for every baseline packet, which leads to reduced bandwidth consumption. However, the channel would be reserved for the transmission of the cargo, preventing any other devices from transmitting while the cargo is being transmitted.

In the setting with cargo, AD again consists of the 22-bit Janus header and F, but, additionally, the first byte of the ADB is considered as associated data, as it provides meta-information about the packet. Since the first byte of the ADB is used for the cargo specification, 26 bits are left in it for user data. Thus, the 26 MSBs of N can be put here, while the 6 LSBs are transmitted as cargo. Following N, the encrypted

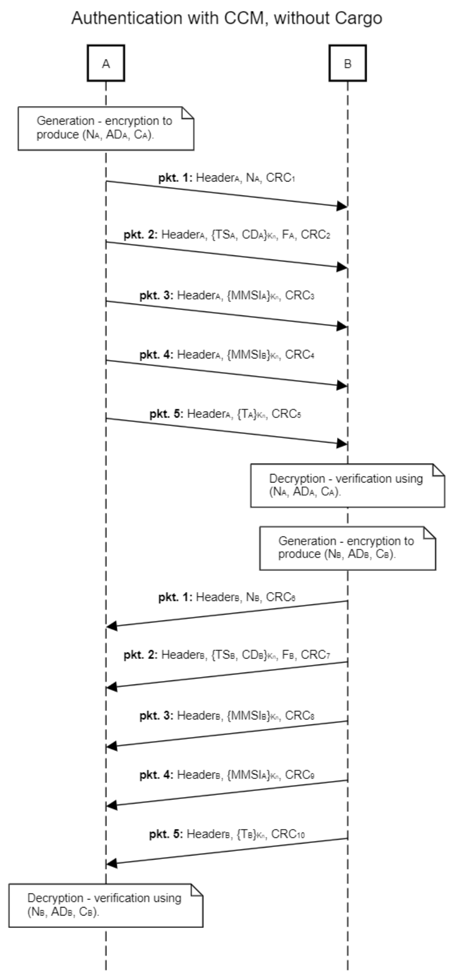
Authentication with CCM, without Cargo
A
B
Generation - encryption to produce (N_A, AD_A, C_A).
pkt. 1: $Header_A$, N_A, CRC_1
pkt. 2: $Header_A$, $\{TS_A, CD_A\}_{K_n}$, F_A, CRC_2
pkt. 3: $Header_A$, $\{MMSI_A\}_{K_n}$, CRC_3
pkt. 4: $Header_A$, $\{MMSI_B\}_{K_n}$, CRC_4
pkt. 5: $Header_A$, $\{T_A\}_{K_n}$, CRC_5
Decryption - verification using (N_A, AD_A, C_A).
Generation - encryption to produce (N_B, AD_B, C_B).
pkt. 1: $Header_B$, N_B, CRC_6
pkt. 2: $Header_B$, $\{TS_B, CD_B\}_{K_n}$, F_B, CRC_7
pkt. 3: $Header_B$, $\{MMSI_B\}_{K_n}$, CRC_8
pkt. 4: $Header_B$, $\{MMSI_A\}_{K_n}$, CRC_9
pkt. 5: $Header_B$, $\{T_B\}_{K_n}$, CRC_{10}
Decryption - verification using (N_B, AD_B, C_B).

Figura 2. CCM used in authentication protocol, without usage of cargo.

payload is transmitted, consisting of the ciphertexts of TS, CD, the two MMSIs, and T. The protocol can be seen in Fig. 3.

For the case of one packet of this format, the total amount of data is $30 + 26 + 8 + 6 + 2 + 29 + 3 + 30 + 30 + 32 = 196$ bits, with the cargo consisting of the last 132 bits. With the data rate of 80 bps provided by Janus, the theoretical encoding time for the entire packet is $196/80 = 2{,}45$ s. For the cargo alone, the encoding time is $132/80 = 1{,}65$ s. This means that at least 1,65 s of channel reservation time must be specified in the first byte of the ADB. According to the lookup table for channel reservation used by Janus, the minimum time that can be reserved with the reservation bits in the ADB to accommodate this cargo length is $\approx 1{,}79$ s. Compared to the encoding time for one baseline packet alone, $64/80 = 0{,}8$ s, this packet format does not impose a substantial overhead with respect to the added benefits of both confidentiality and integrity of all data needed to perform message authentication, entity authentication, and session key establishment.

III-B. Authenticated encryption with AEGIS

A potentially even more suitable algorithm than CCM is AEGIS, proposed by Wu and Preneel [1]. Unlike CCM, which is a special mode of operation for a block cipher, AEGIS is a dedicated authenticated encryption algorithm, which also employs an underlying block cipher, but uses a message to

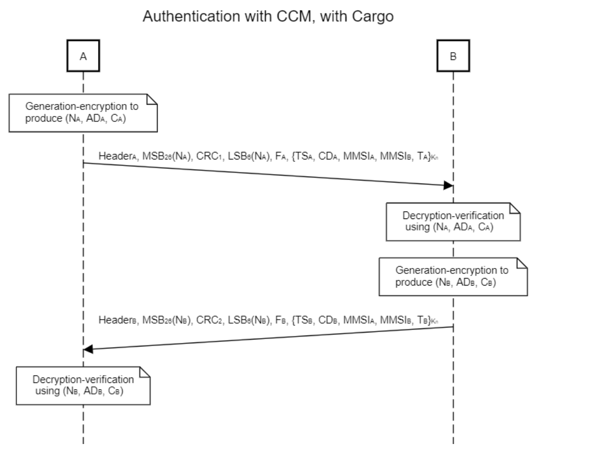
Authentication with CCM, with Cargo
A
B
Generation-encryption to produce (N_A, AD_A, C_A)
$Header_A$, $MSB_{26}(N_A)$, CRC_1, $LSB_6(N_A)$, F_A, $\{TS_A, CD_A, MMSI_A, MMSI_B, T_A\}_{K_n}$
Decryption-verification using (N_A, AD_A, C_A)
Generation-encryption to produce (N_B, AD_B, C_B)
$Header_B$, $MSB_{26}(N_B)$, CRC_2, $LSB_6(N_B)$, F_B, $\{TS_B, CD_B, MMSI_A, MMSI_B, T_B\}_{K_n}$
Decryption-verification using (N_B, AD_B, C_B)

Figura 3. CCM used in authentication protocol, with usage of cargo.

update the state of the cipher. As with CCM, AES is most commonly used as the underlying block cipher and we shall also describe this version for our purposes.

For consistency with other relevant solutions, such as Venilia, we focus on the 256-bit key version, namely AEGIS-256, which utilizes a 256-bit key and 256-bit IV, operates on 16-byte message blocks and uses 6 AES round functions.

AEGIS-256 defines a state update function, which updates an internal state using 6 iterations of AES. The main operations are the initialization, processing of the associated data, encryption, and finalization. Each of the operations updates the state and uses it to perform their respective tasks. The tag T can be of any length up to 128, which is the AES block size.

III-B1. Argumentation for the use of AEGIS: AEGIS was one of the winning submissions to the CAESAR contest for authenticated encryption algorithms. Hence, we believe that an underwater solution based on AEGIS will keep underwater security schemes up to date with their counterparts above water, even if CCM gets discontinued in WiFi networks. In addition, AEGIS has already been deployed in autonomous vessels that apply the Robot Operating System (ROS) [12]. Since the underwater environment has similar devices and characteristics as the surface vessels using ROS, we believe the application of AEGIS under water is a natural next step.

Wu and Preneel provide a security analysis of AEGIS in addition to its specification. They name three requirements for the secure operation of AEGIS:

1. Each key K used for initialization must be generated uniformly at random.
2. An IV must not be used more than once during the lifetime of a key and each key and IV pair must only be used with one size of the authentication tag T.
3. If the verification of the tag T fails, the decrypted plaintext and wrong T must not be disclosed to the public.

III-B2. Application in Janus-Based Communication: The protocol for authentication with AEGIS is very similar to the CCM version, described above, both with and without Janus cargo packets. The same values are transmitted in both schemes, resulting in an equal amount of bits. The nonce N from CCM is denoted IV here, but these values serve a very

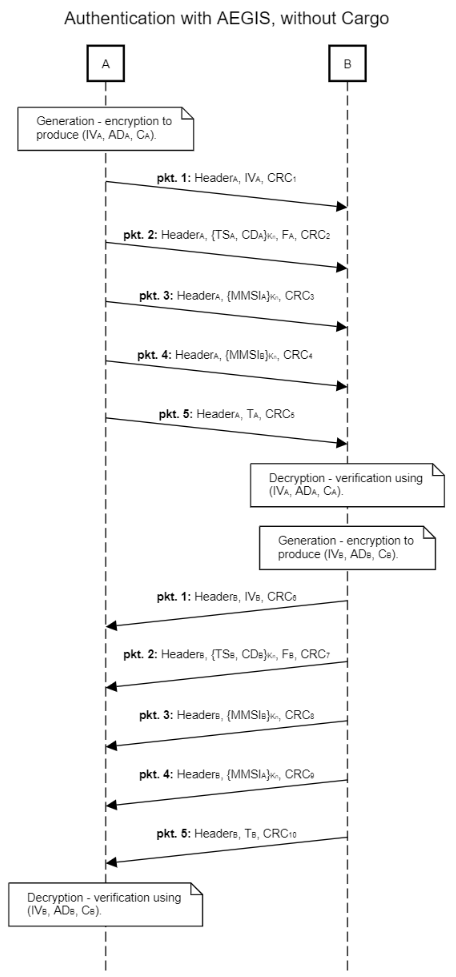
Authentication with AEGIS, without Cargo
A
B
Generation - encryption to produce (IV_A, AD_A, C_A).
pkt. 1: $Header_A$, IV_A, CRC_1
pkt. 2: $Header_A$, $\{TS_A, CD_A\}_{K_n}$, F_A, CRC_2
pkt. 3: $Header_A$, $\{MMSI_A\}_{K_n}$, CRC_3
pkt. 4: $Header_A$, $\{MMSI_B\}_{K_n}$, CRC_4
pkt. 5: $Header_A$, T_A, CRC_5
Decryption - verification using (IV_A, AD_A, C_A).
Generation - encryption to produce (IV_B, AD_B, C_B).
pkt. 1: $Header_B$, IV_B, CRC_6
pkt. 2: $Header_B$, $\{TS_B, CD_B\}_{K_n}$, F_B, CRC_7
pkt. 3: $Header_B$, $\{MMSI_B\}_{K_n}$, CRC_8
pkt. 4: $Header_B$, $\{MMSI_A\}_{K_n}$, CRC_9
pkt. 5: $Header_B$, T_B, CRC_{10}
Decryption - verification using (IV_B, AD_B, C_B).

Figura 4. AEGIS used in authentication protocol, without usage of cargo.

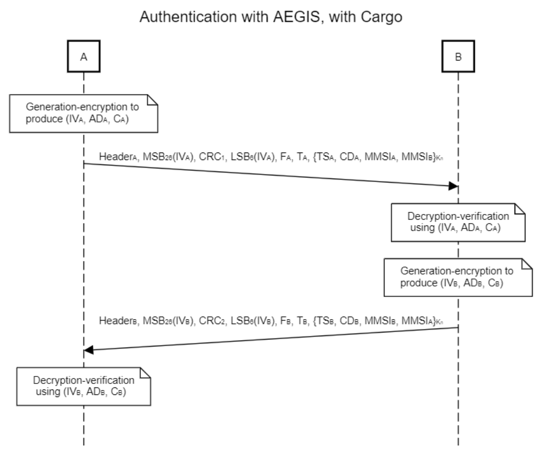
Authentication with AEGIS, with Cargo
A
B
Generation-encryption to produce (IV_A, AD_A, C_A)
$Header_A$, $MSB_{26}(IV_A)$, CRC_1, $LSB_6(IV_A)$, F_A, T_A, $\{TS_A, CD_A, MMSI_A, MMSI_B\}_{K_n}$
Decryption-verification using (IV_A, AD_A, C_A)
Generation-encryption to produce (IV_B, AD_B, C_B)
$Header_B$, $MSB_{26}(IV_B)$, CRC_2, $LSB_6(IV_B)$, F_B, T_B, $\{TS_B, CD_B, MMSI_B, MMSI_A\}_{K_n}$
Decryption-verification using (IV_B, AD_B, C_B)

Figura 5. AEGIS used in authentication protocol, with usage of cargo.

similar purpose. The protocol without cargo can be seen in Fig. 4, while the version with cargo is shown in Fig. 5.

IV. Conclusion

In this paper, we have proposed two mechanisms of authenticated encryption integrated in the Janus-based underwater acoustic communication system. We gave the arguments in favor of these proposals and we believe that they can provide a satisfactory level of security in Janus. We have shown that the two proposed solutions have potential to be accepted for use with Janus when longer messages are to be securely transmitted. We provided the calculations that show that the computational overhead related to these mechanisms is acceptable.

Referencias

[1] Wu H., Preneel B., "AEGIS: A fast authenticated encryption algorithm", *Proceedings of SAC 2013*, pp. 1–21, 2013.
[2] Biryukov A., Kushilevitz E., "Improved cryptanalysis of RC5", *Proceedings of EUROCRYPT'98*, pp. 85–99, 1998.
[3] Domingo M., "Securing underwater wireless communication networks", *IEEE Wireless Communications*, Vol. 18, No. 1, pp. 1–21, 2011.
[4] Ghannadrezaii H., Bousquet J., "Securing a Janus-based flooding routing protocol for underwater acoustic networks", *Proceedings of OCEANS 2018*, pp. 1–7, 2018.
[5] Potter J., Alves J., Green D., Zappa G., Nissen I., McCoy K., "The JANUS underwater communications standard", *2014 Underwater Communications and Networking (UComms)*, pp. 1–4, 2014.
[6] Jonsson J., "On the security of CTR + CBC-MAC", *Proceedings of SAC 2003*, pp. 76–93, 2003.
[7] Dworkin M., "Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality", NIST Special Publication 800-38C, 2004.
[8] Petrioli C., Saturni G., Spaccini D., "Feasibility study for authenticated key exchange protocols on underwater acoustic sensor networks", *Proceedings of WUWNET '19*, pp. 1–5, 2019.
[9] Téglásy B., Wengle E., Potter J., Katsikas S., "Authentication of underwater assets", paper in preparation, 2022.
[10] Hobbs A., Holdcroft S., "Tiny Underwater Block cipher (TUBcipher): 27-bit encryption scheme for JANUS class 17", *Dstl Cyber and Information Systems*, 2021.
[11] Hobbs A., Holdcroft S., "JANUS class 17 Venilia: Secure pre-canned messaging", *Dstl Cyber and Information Systems*, 2021.
[12] Volden Ø., Solnør P., Petrović S., Fossen T., "Secure and efficient transmission of vision-based feedback control signals", *Journal of Intelligent & Robotic Systems*, Vol. 103, No. 2, 2021.
[13] Whiting D., Housley R., Ferguson N., "AES encryption & authentication using CTR mode & CBC-MAC", IEEE 802.11-02, 2002.