UNIVERSITÀ DI PISA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Master Degree in CyberSecurity

Interoperable maritime messaging architecture using VDES and SECOM security standards

Supervisor:

Prof: Stefano Chessa

Prof: Paolo Pagano

Candidate:

Nicola Lepore

ACADEMIC YEAR 2024/2025

Contents

List of Figures

Chapter 1

Introduction

1.1 Maritime Digitalisation Context

During the past two decades, the maritime sector has undergone a progressive digitalisation process that has profoundly transformed the way ships, ports, and authorities interact with each other. The introduction of digital communication systems, intelligent sensors, and data management platforms has enabled increased efficiency in port operations, improved navigation safety, and paved the way for new information-based maritime services.

Vessel Traffic Services (VTS), coastal and satellite monitoring systems, and e-Navigation platforms promoted by the International Maritime Organization (IMO) [1] and the International Association of Marine Aids to Navigation and Lighthouse Authorities (IALA) [2] have made the availability of reliable and timely data central. In this scenario, protocols such as the Automatic Identification System (AIS) [3] and, more recently, the VHF Data Exchange System (VDES) [4, 5] have played a fundamental role, allowing the exchange of operational safety and environmental information between ships and coastal stations.

In parallel, the international community has recognised the need to introduce specific cybersecurity standards for the maritime domain, capable of protecting transmitted data against interception, manipulation, and targeted attacks. In this regard, the SECOM (Secure Communication) standard, part of the IEC 63173-2 framework [6], represents an important step toward an ecosystem of secure and interoperable digital maritime communications, guaranteed through an identity infrastructure based on the Maritime Connectivity Platform (MCP) [7].

The overall objective of this process is twofold: on the one hand, to ensure safe and resilient operations even in the presence of increasingly sophisticated cyber threats; on the other hand, to facilitate the transition toward an integrated digital maritime environment capable of supporting innovations such as autonomous navigation, advanced environmental monitoring, and intelligent port services.

1.2 Current Issues in Maritime Communications

Despite the progress made, maritime communications today face a series of structural and operational challenges that limit their reliability and security.

The first issue concerns the lack of authentication in AIS and VDES messages. AIS was designed with the objective of ensuring the availability and dissemination of information rather than its security [3]. Consequently, messages are subject to spoofing, i.e., they can be manipulated or falsified by an attacker wishing to deceive a ship or a shore-based node. AIS spoofing incidents demonstrate how it is possible to create phantom ships, alter transmitted routes, or obscure the position of a real vessel, with potentially serious consequences both operationally and in terms of maritime safety.

The second issue relates to compatibility with legacy systems. AIS, widely deployed globally and integrated into numerous port and navigation infrastructures, uses a fixed and very limited packet format (only a few bytes are available for data) [3]. This makes it difficult, if not impossible, to directly integrate digital signatures or complex authentication mechanisms without compromising backward compatibility with existing equipment [2].

In addition, there are difficulties arising from interoperability between IP and non-IP systems. Although modern protocols such as SECOM operate on IP stacks and allow the use of advanced cryptographic primitives, many traditional maritime systems (e.g. AIS via VHF) use dedicated radio channels with reduced capacity, which are difficult to integrate into IP networks. The gap between these two realities complicates the adoption of a unified standard and creates the risk of technological fragmentation.

Finally, challenges also arise from the management of digital identities and certificates, latency in data transmission, limited scalability of some approaches, and divergences among the directives issued by different regulatory bodies (IMO, IALA, and the European Union), each pursuing distinct objectives. As a result, complying with the specific requirements of each domain represents a significant challenge, making it difficult to define an architecture that is interoperable, secure, and simultaneously adoptable on a large scale.

1.3 Thesis Objectives

This thesis aims to address the challenges described above through the design and prototyping of a secure and interoperable maritime messaging architecture, based on the integration between VDES communication channels [4] and the SECOM security framework [6].

The specific objectives are to:

- Integrate non-IP communications (particularly AIS and ASM (Application Specific Messages) messages transmitted via VHF) with secure IP communications managed by SECOM, ensuring interchangeability and consistency of security mechanisms.

- Develop a mapping layer capable of using MCP [7] for communication via VDES while maintaining compatibility with legacy AIS systems.
- Implement and test the TESLA protocol [8, 9] as a lightweight and efficient authentication mechanism, capable of reducing overhead compared to digital signatures and adapting to the bandwidth constraints of VHF channels.
- Create a simulation and prototypal validation testbed that can emulate realistic communication scenarios between ships, ports, and authorities.
- Evaluate the proposed architecture according to security, interoperability, and performance metrics in order to provide useful recommendations for future IMO/IALA standardisation.

1.4 Thesis contribution

The main expected contributions of this research are the following:

- Design of an interoperable architecture that integrates VDES and SECOM, enabling the use of advanced security mechanisms even for non-IP communications.
- Experimental demonstration of the feasibility of TESLA as an authentication scheme for AIS and ASM, with a detailed analysis of its impact in terms of overhead, latency, and robustness.
- Contribution to the scientific community and standardisation bodies (IMO, IALA), providing practical recommendations and useful results for the adoption pathway of maritime digital standards.

Chapter 2

Background and State of the Art

2.1 Maritime Communications Architecture (AIS, ASM, VDES)

The maritime communications system has been historically designed to ensure navigation safety and situational awareness through the constant exchange of information between ships and coastal infrastructure. The Automatic Identification System (AIS), made mandatory by the IMO for ships of certain sizes starting in 2004, represents the cornerstone of this architecture [1].

AIS operates primarily in VHF broadcast mode, transmitting data such as position, speed, course, ship identity, and service messages. Despite its success and widespread adoption, AIS is limited by the fixed structure and constrained capacity of its packets. The payload sizes, defined by the ITU-R M.1371[3] standard, vary: common position reports (Types 1-3) use approximately 168 bits, but more complex safety messages (such as Type 12) can aggregate data exceeding 1000 bits, occupying multiple slots. However, this architecture, optimised for broadcast transmission of specific data, prevents its natural extension to complex security functionalities (such as PKI digital signatures) due to format constraints and per-slot capacity limitations.

To address these needs, the VHF Data Exchange System (VDES) was developed, which extends AIS by introducing three main channels [4]:

- **AIS channels**: compatible with the legacy standard.
- **ASM (Application Specific Messages)**: for sending additional data, such as meteorological, hydrographic, or service information.
- **VHF Data Exchange (VDE) channels** (VDE-TER for terrestrial communications and VDE-SAT for satellite communications): designed for higher-capacity and long-range data transmissions.

The adoption of VDES therefore aims to enable complete digitalisation of maritime services, providing a scalable infrastructure capable of supporting e-Navigation. However, while VDES expands transmission capabilities, it does not natively resolve the issue of message security, which remains delegated to additional protocols and architectures [5].

2.2 SECOM (IEC 63173-2) and the Maritime Connectivity Platform (MCP)

In parallel with the evolution of maritime radio systems, the maritime world has developed standards for IP-based communication security. Among these, one of the most relevant is the SECOM (Secure Communication) standard defined by the IEC 63173-2 specification [6].

SECOM provides a framework for secure transfer of S-100 data (the common standard for electronic navigation data) through IP protocols. The main features of SECOM are:

- Use of signed and encrypted JSON payloads, ensuring authenticity, integrity, and confidentiality.
- Modular structure, compatible with modern web services (e.g., REST APIs).
- Integration with digital identity management systems.

The identity system is enabled by the Maritime Connectivity Platform (MCP), a platform that allows the creation, discovery, and validation of digital certificates for ships, ports, and authorities [7]. One of its central components is the Maritime Identity Registry (MIR), which serves as a distributed registry of digital identities. MCP simplifies the distribution and management of certificates, allowing ships and authorities to quickly establish secure sessions without the need for complex infrastructures.

The combination of SECOM for secure data transport and MCP/MIR for identity management today represents the most solid foundation for building secure digital maritime services over IP [6, 7]. However, the challenge remains to extend these mechanisms to non-IP systems, such as AIS and ASM via VHF, to achieve true end-to-end interoperability.

2.3 Security in Maritime Communications

Modern maritime communications must face a rapidly evolving threat landscape. The main risk categories include:

- **Spoofing**: manipulation of AIS/VDES messages to generate phantom ships or alter existing routes.

- **Replay attacks**: reuse of valid messages to deceive receiving systems.
- **Jamming**: intentional disruption of VHF radio channels, compromising availability.
- **Man-in-the-middle**: interception and modification of messages during transmission.

In this context, standards require that every communication architecture guarantee at least three fundamental properties [6]:

1. **Authenticity**: the message must actually originate from the declared source.
2. **Integrity**: data must not be altered during transmission.
3. **Confidentiality**: content must remain accessible only to authorised actors.

To these is added the requirement of interoperability, i.e., the ability of systems from different generations and technologies (IP and non-IP) to communicate with each other without loss of functionality or security.

2.4 Literature Review and Contextualisation

The recent literature on maritime communication security converges in several main research directions. A first line of work focuses on the development of Digital Maritime Services (DMS), where the transition toward integrated e-Navigation platforms repeatedly exposes the limitations of the current AIS ecosystem, particularly its rigidity and lack of natively embedded security mechanisms. Parallel studies investigate secure transport layers, assessing the applicability of TLS, SECOM, and related IP-based frameworks to maritime services [6]. Additional research examines the progressive adoption of VDES as the natural evolution of AIS, analysing its extended capacity, backward compatibility requirements, and emerging operational use cases [4].

A further stream of contributions evaluates security models based on SECOM, validating the effectiveness of signed and encrypted JSON payloads while emphasising their difficulty of deployment over non-IP channels, where bandwidth and frame-structure constraints impose severe limitations. Complementary work compares authentication mechanisms grounded in PKI and TESLA [9]. While digital signatures provide strong cryptographic assurances, they incur non-negligible overhead, making them unsuitable for high-frequency or bandwidth-constrained transmissions. In contrast, TESLA is consistently identified as a lightweight, scalable, and quantum-resistant alternative tailored for broadcast environments [8].

Across these research efforts, a clear consensus emerges: achieving interoperability between IP-based and non-IP maritime communication systems requires a mapping layer capable of transporting SECOM security mechanisms over VDES channels. This layer must maintain efficiency constraints while remaining fully compatible with legacy infrastructures.

This observation is strongly aligned with the material presented in the official *IALA Guideline G1192 – VDES Authentication Techniques, Edition 1.0 (June 2025)* [2], from which the technical excerpt in this chapter is derived. The guideline provides an authoritative analysis of the structural barriers that currently prevent the widespread adoption of interoperable authentication standards across AIS, VDES, and related systems. It identifies four main categories of constraints intrinsic to the existing ecosystem: the immutability of AIS message formats, the extremely limited bandwidth of VHF channels, the operational challenges of global maritime PKI deployment, and the regulatory divergence between international standardisation bodies.

These barriers directly reinforce the conclusions of the literature: high-overhead cryptographic schemes cannot be seamlessly integrated into AIS or low-capacity VDES channels, and authentication mechanisms must therefore be both backward-compatible and computationally lightweight. The guideline explicitly highlights TESLA-like approaches as suitable candidates, given their ability to operate without modifying message structures and with minimal bandwidth consumption [2, 10].

In this thesis, these insights form the foundation for the design choices presented in subsequent chapters. The proposed solution implements the constraints and recommendations described in the guideline, demonstrating that efficient authentication can be achieved while remaining compliant with legacy, performance, and interoperability requirements identified by both the academic literature and the IALA Guideline G1192 [2, 10].

Chapter 3

Cryptographic Background of the Protocol

The security and integrity of communications within the proposed protocol depended on a hybrid cryptographic architecture that leverages the complementary strengths of symmetric-key and asymmetric-key (public-key) cryptography. A solid understanding of these two paradigms is essential for analysing the specific role played by each cryptographic component (such as hash functions, Message Authentication Codes (MACs), and digital certificates) in ensuring confidentiality, integrity, authentication, and non-repudiation within the VDES domain.

3.1 Introduction to Asymmetric and Symmetric Cryptography

3.1.1 Symmetric-Key Cryptography

Symmetric-key cryptography represents the earliest and most widely deployed form of encryption and is based on the use of a *single secret key* (K_{sec}) shared in advance between communicating entities. This key governs both the encryption and decryption processes, which can be formally expressed through the relation

$$D(K_{sec},\, E(K_{sec},\, M)) = M,$$

where M denotes the plaintext, $E(K_{sec}, M)$ the ciphertext, and K_{sec} the shared secret. Due to their intrinsic computational efficiency, symmetric algorithms are particularly suited for high-throughput scenarios or environments with constrained resources and strict latency requirements, conditions frequently encountered in maritime radio communication systems [11]. Their performance advantage, often ranging from two to three orders of magnitude faster than asymmetric mechanisms, makes them indispensable for protecting large volumes of data. However, their main limitation concerns the secure distribution of K_{sec}: each pair of communicating nodes must first establish a protected channel to exchange the key, a requirement that does not scale well in dynamic and globally distributed networks such as those in the maritime domain. In the protocol described in this work, symmetric primitives primarily implemented through Message Authentication Codes (MACs) are employed to ensure the integrity and authentication of high-frequency data flows, including VDES payload transmissions.

3.1.2 Asymmetric-Key Cryptography

Asymmetric-key cryptography, or public-key cryptography, was introduced to overcome the key distribution challenges inherent to symmetric systems. It relies on a pair of mathematically related, yet distinct keys: a *public key* (K_{pub}), which may be freely shared, and a *private key* (K_{priv}), which must remain confidential [12]. The public key enables the encryption of messages intended for the private-key holder or the verification of digital signatures produced with K_{priv}, while the private key is used to decrypt messages encrypted under K_{pub} or to generate signatures that authenticate the sender. The security of these schemes relies on the computational infeasibility of deriving K_{priv} from K_{pub}, which allows secure communication without requiring a prior exchange of secrets. Despite their strong security properties, asymmetric algorithms such as RSA and Elliptic Curve Cryptography (ECC) incur significantly higher computational costs compared to symmetric approaches. For this reason, they are not typically used to encrypt bulk data; instead, they secure the exchange of symmetric session keys, where a short K_{sec} is encrypted asymmetrically before being used for efficient symmetric encryption, and provide digital signatures and non-repudiation. In this process, a message digest is signed with K_{priv}, allowing any party possessing K_{pub} to verify the authenticity and integrity of the communication [13].

3.2 Cryptographic Hash Functions

Cryptographic hash functions are fundamental components of any modern security protocol. They act as compact, non-reversible digital fingerprints of input data of arbitrary length, providing strong guarantees of message integrity [14].

3.2.1 Definition and Properties of an Ideal Hash Function

A *cryptographic hash function* (H) is a deterministic mathematical function that accepts an input message M of arbitrary length and produces a fixed-length output string h, known as the *hash value* or *digest* ($h = H(M)$).

TTo be considered secure and suitable for use in authentication and integrity mechanisms, such as those required by the TESLA protocol, a cryptographic hash function must satisfy the following three core resistance properties [11]:

Preimage Resistance (One-Way Property) Given an output value h, it must be *computationally infeasible* to recover any input message M such that:

$$H(M) = h.$$

This property is critical for protecting the time-delayed keychains used within the protocol.

Second Preimage Resistance (Weak Collision Resistance) Given an input message M_1, it must be *computationally infeasible* to find a different message M_2 such that:

$$H(M_1) = H(M_2) \quad \text{with } M_2 \neq M_1.$$

This ensures that an attacker cannot modify a message while preserving a known digest, thereby upholding integrity.

Collision Resistance (Strong Collision Resistance) It must be *computationally infeasible* to find *any* pair of distinct messages (M_1, M_2) such that:

$$H(M_1) = H(M_2), \quad M_1 \neq M_2.$$

Collision resistance is essential for the security of digital signatures and certificates. Its practical difficulty is influenced by the *Birthday Paradox*: for an output of length L bits, a collision can be found with a complexity of approximately $O(2^{L/2})$ rather than $O(2^L)$. Consequently, the output length must be large enough to ensure meaningful security.

3.2.2 The SHA-256 Algorithm

SHA-256 (Secure Hash Algorithm, 256-bit) is one of the most widely adopted cryptographic hash functions and belongs to the **SHA-2 family**, standardised by NIST [14].

Overview and Structure

SHA-256 is based on the **Merkle-Damgård** construction. The hashing process consists of three main stages:

1. **Padding:**

   The input message (M) is extended with additional bits so that its final length is exactly divisible by 512 bits.

   The padding procedure works as follows: first, a single bit set to '1' is added; then a sequence of '0' bits is added; finally, a 64-bit value encoding the original length of the message (before padding) is added.

2. **Initialisation:**

   The algorithm initialises an internal state of 256 bits, consisting of eight 32-bit working registers $(H_0^{(0)}, H_1^{(0)}, \ldots, H_7^{(0)})$. These are set to predefined constants derived from the fractional parts of the square roots of the first eight prime numbers.

3. **Compression Function:**

   The padded message is processed in 512-bit blocks. Each block is passed through a **64-round compression function** that mixes:

   - bitwise logical operations,
   - modular additions,
   - right rotations,
   - and 64 constant values derived from the cube roots of the first prime numbers.

   The output of each round updates the intermediate hash state, which becomes the input of the next block.

The final digest is obtained from the state produced after processing the last message block.

Output Size and Security Considerations

- **Output Length:** SHA-256 produces a **256-bit** (32-byte) digest.
- **Security Level:**

  A hash function with L-bit output provides $L/2$ bits of collision resistance due to birthday attacks. Thus, SHA-256 provides a security strength of **128 bits** against collision search.
- **Implications for the Protocol:**

  The 256-bit output of SHA-256 ensures that brute-force collision search (complexity $O(2^{128})$) is computationally infeasible with foreseeable technology [14].

  This makes SHA-256 suitable for use in TESLA and in HMAC constructions, where the integrity and non-reversibility of the key chain depend on the underlying hash function.

3.3 Message Authentication Codes (MAC)

Message Authentication Codes (MACs) are symmetric-key cryptographic functions designed to provide guarantees of message integrity and origin authentication. They are essential in protocols such as TESLA, where the integrity of data blocks and their unambiguous attribution to a legitimate sender are critical requirements [8].

3.3.1 Purpose of Authentication and Integrity Verification

Within the context of VDES communications and delayed-disclosure security protocols, MACs fulfil two primary objectives:

Message Integrity:

A MAC ensures that the received message (M) has not been altered, modified, truncated, or reordered by an adversary during transmission. Upon reception, the verifier recomputes the MAC and compares it with the transmitted value; any mismatch indicates tampering.

Origin Authentication:

Since MAC generation requires a secret key (K) shared exclusively between the sender and the legitimate receiver, a valid MAC demonstrates that the message was produced by the only other entity possessing K. This prevents unauthorised parties from injecting forged messages.

Unlike digital signatures, which rely on asymmetric cryptography and additionally provide non-repudiation, MACs operate in the symmetric-key domain. This makes them extremely fast to compute and verify, a property that is indispensable for processing continuous packet flows such as VDES data streams. However, because the sender and receiver share the same key, MACs do not provide non-repudiation.

3.3.2 HMAC Algorithm (Hash-based Message Authentication Code)

HMAC is a widely used MAC construction defined in RFC 2104. Using a cryptographically secure hash function (H,in this protocol SHA-256) in combination with a secret key K to generate an authentication tag, while mitigating weaknesses that would arise from using the key directly within the hash function [15].

Structure and Operational Mechanism

The HMAC output, denoted as the authentication tag T, is computed as:

$$T = \text{HMAC}_K(M) = H\big((K \oplus \text{opad}) \mid\mid H((K \oplus \text{ipad}) \mid\mid M)\big)$$

Where:

- M is the input message.
- K is the secret key (of length L bits).
- B is the internal block size of the hash function ($B = 512$ bits for SHA-256).
- `ipad` is a repeated sequence of the byte `0x36` of length B bytes (inner padding).
- `opad` is a repeated sequence of the byte `0x5C` of length B bytes (outer padding).

- $||$ denotes concatenation.
- $\oplus$ denotes the bitwise XOR operation.

The computation proceeds in two sequential hashing steps:

- **Inner Hash:**

  The key K is XORed with `ipad` and concatenated with the message M. The resulting sequence is processed by the hash function H:

  $$H_{\text{inner}} = H\big((K \oplus \text{ipad}) \mid\mid M\big)$$

- **Outer Hash:**

  The key K is XORed with `opad` and concatenated with the inner hash output H_{inner}. The result is hashed again to produce the final authentication tag:

  $$T = H\big((K \oplus \text{opad}) \mid\mid H_{\text{inner}}\big)$$

This two-layer construction strengthens the security of the MAC, ensuring robustness against key-recovery and extension attacks while maintaining high computational efficiency, a crucial property for real-time and high-frequency communication channels such as VDES [15].

3.4 Public Key Infrastructure (PKI) and Digital Certificates

Security management within the maritime domain requires mechanisms for unique identification and trust verification among distant and previously unknown entities. The resolution of this authentication problem, and the reliable association between a public key and a real identity, is provided by the Public Key Infrastructure (PKI) [13].

3.4.1 Digital Identity and Trust

A *Digital Identity* is the set of attributes and credentials that uniquely identify a person, device, or service within a telematic system. In a cryptographic context, the fundamental challenge lies in securely binding a public key (used for signature verification and asymmetric encryption) to a physical or logical identity (e.g., a vessel, a port authority, or a communication service).

Trust in asymmetric communication is not inherent; it must be established. A PKI introduces a universally recognised trusted third party, the Certificate Authority (CA), which attests to the authenticity of the binding between identity and public key. Trust is therefore delegated: instead of relying directly on the public key of an unknown entity, a system trusts the CA that certified it. This model is crucial for protocol scalability, allowing any entity to authenticate another regardless of their location, provided that both recognise the same root CA.

3.4.2 Structure and Contents of X.509 Certificates

The most widely adopted standard for digital certificates is X.509 [13]. A digital certificate is essentially a document that contains the public key of an entity, its identifying information, and a digital signature generated by the CA.

An X.509 certificate includes several critical fields, such as:

- **Version:** Indicates the version of the X.509 standard used.
- **Serial Number:** A unique identifier assigned by the CA.
- **Signature Algorithm:** Specifies the algorithm used by the CA to sign the certificate (e.g., `SHA-256 with RSA`).
- **Subject:** The identity (e.g., name, organisation, locality) of the public-key holder.
- **Subject Public Key:** The certified public key, including the cryptographic algorithm (e.g., RSA, ECC) used to generate it.
- **Issuer:** The distinguished name (DN) of the CA that issued and signed the certificate.
- **Validity:** The start and end dates during which the certificate is considered valid.
- **Extensions:** Optional fields (e.g., *Key Usage*, *Subject Alternative Name* - SAN) that specify the intended usage of the certified key (e.g., encryption, digital signature, client authentication).

3.4.3 Certificate Authorities (CA) and the Chain of Trust

The Certificate Authority (CA) is responsible for issuing, revoking, and managing certificates. Its main role is to validate the identity of the requester before binding and signing the associated public key.

The PKI trust model is hierarchical and relies on a *Chain of Trust* [13]:

- **Root CA:** The highest and most trusted entity in the hierarchy. Its certificate is self-signed and must be distributed and pre-installed on all systems and devices (e.g., VDES units or shore-side infrastructure) participating in the communication environment.

- **Intermediate CAs:** For operational and security reasons, the Root CA does not directly sign end-entity certificates. Instead, it delegates signing authority to one or more intermediate CAs.
- **End-Entity Certificates:** Certificates are issued to individual users, vessels, or services.

When a system receives a certificate, its validation software verifies the signature of each certificate in the chain up to the pre-installed Root CA. If all signatures are valid and none of the certificates have expired or been revoked, trust in the end-entity certificate is established.

3.4.4 Use of Certificates in the Maritime Domain

The maritime environment requires a specialised PKI to manage the heterogeneity and mobility of the actors involved.

Certificates and Maritime Identities (Maritime Resource Name - MRN)

To ensure global interoperability, maritime resource identities are standardised through the *Maritime Resource Name* (MRN) scheme [2].

The MRN is a uniform identification system for digital and physical maritime resources (e.g., a VDES unit, a navigation software component, or a shipping company).

X.509 certificates used in the maritime domain typically include the MRN as the primary identifier in the *Subject Alternative Name (SAN)* field or within the *Subject* field (according to e-Navigation regulatory specifications). This direct association between the public key and the MRN ensures that a certificate can unambiguously authenticate the origin of a message, enabling receiving systems to authorise actions based on the type and role of the identified maritime resource.

Role of the Maritime Connectivity Platform (MCP) PKI

The Maritime Connectivity Platform (MCP) is a foundational infrastructure for maritime digitalisation. The PKI operated by MCP provides the designated trust framework for authenticating services and users within this ecosystem.

The MCP PKI functions as the hierarchical CA infrastructure that issues and manages certificates for all entities participating in advanced e-Navigation and VDES services [7].

Chapter 4

Requirements and Problem Analysis

4.1 Limitations of AIS/ASM Packets and VHF Channels

As previously discussed in Section 2.1, one of the main obstacles to the introduction of advanced security mechanisms in maritime communications is represented by the structural rigidity and payload constraints of AIS packets. Although the ITU-R M.1371 standard allows specific messages (e.g., Type 5 or 12) to extend over multiple slots exceeding 1000 bits, the most common and critical messages for navigation, such as position reports (Types 1 and 3), are allocated in single slots with a fixed payload of 168 bits, distributed across rigidly specified format fields [3].

Even Application Specific Messages (ASM), while offering greater flexibility, remain constrained by the overall capacity of the VHF channel. Consequently, direct insertion of modern asymmetric cryptographic primitives, such as digital signatures, within these predefined formats is impractical. Such an approach would result in an unsustainable use of transmission capacity and, most importantly, would break compatibility with legacy receivers [2].

This limitation is not only technical but also regulatory: IMO and IALA require that the AIS ecosystem remain backward compatible to ensure the continuity of the global service [1]. This makes it impossible to substantially modify the AIS format, requiring an alternative approach, such as using a parallel channel (VDE-TER) to transmit authentication information [4].

4.2 Need for Interoperability Between IP and Non-IP Communications

The maritime communications architecture is currently defined by a problematic coexistence of two fundamentally different technological paradigms:

1. **IP-based systems**: Modern architectures such as SECOM (IEC 63173-2) and the Maritime Connectivity Platform (MCP). These systems operate on standard IP stacks (leveraging terrestrial, satellite, or 4G/5G networks) and are designed for end-to-end (E2E) digital services. Their security is based natively on public-key cryptography, digital signatures, and flexible payloads such as JSON [6, 7].

2. **Traditional non-IP systems (radio-based)**: Protocols such as AIS and the ASM channels of VDES, operating over VHF. These are broadcast-centric systems, based on time slots (TDMA), designed for short/medium-range situational awareness and constrained to rigid packet formats [4].

This dichotomy is not merely a technical distinction, but generates a fundamental **security gap** when a digital service (e.g., an S-100 chart update defined by SECOM) must transit from the IP domain (shore-based authority) to the non-IP domain (ship via VHF). The E2E security guarantees provided by SECOM, the authenticity and integrity ensured by a digital signature, are not natively translatable into the rigid formats of AIS/ASM [2].

This generates a series of critical issues:

- **Interruption of the chain of trust**: System fragmentation is not only inefficient, but also interrupts the chain of trust. A secure and signed payload in the IP world must be "unpacked" and converted into an insecure (unauthenticated) non-IP message for VHF transmission, exposing it to manipulation (spoofing) precisely in the last mile.

- **Fundamental security misalignment**: A blatant inconsistency occurs. The same data (e.g., the position of a hazard) are considered "trustworthy" if received via SECOM (IP) but "untrustworthy" (unauthenticated) if received via AIS/ASM. This forces authorities to manage parallel technological silos, increasing operational complexity, and the risk of decisions based on inconsistent or unverified data.

Interoperability, therefore, does not simply mean "translating formats". It requires, as mentioned previously, the design of a mapping layer that acts as a **security bridge**. This layer must be capable of extracting and preserving the cryptographic properties of the SECOM payload (authenticity, integrity) and transporting them through the VDES channel; for example, by decoupling the data from its authentication mechanism (such as TESLA) without altering the backward compatibility required for legacy AIS messages [8].

4.3 Legacy System Compatibility Constraints

The principle of backward compatibility is not a simple technical requirement, but the primary economic and operational constraint that shapes any evolution in the

AIS domain. The global installed base of AIS equipment, which includes millions of transponders on SOLAS and non-SOLAS vessels, coastal stations, and VTS systems, constitutes the fundamental infrastructure for Safety of Life at Sea (SOLAS) services [1].

The technological inertia of this sector is considerable; the life cycles and replacement of onboard equipment span decades. Consequently, any modification to the AIS message format that breaks compatibility (a "breaking change") would instantly render the existing infrastructure obsolete. The prohibitive costs and transition times, incompatible with safety requirements, make this option impractical. Furthermore, port systems and Vessel Traffic Services (VTS) centres have critical operational processes that strictly depend on standardised and stable AIS data flows for their daily traffic monitoring and management functions.

For these reasons, a "substitute" approach is excluded. The strategy adopted in this thesis is the only viable path: an **additive architecture** that integrates authentication mechanisms in a parallel channel (VDE-TER) without altering AIS/ASM packets [2]. This decoupling ensures that the entire legacy ecosystem continues to function without modifications, receiving the traditional data stream, while updated (VDES-enabled) systems can leverage the parallel channel to access security metadata (such as TESLA MACs), thus obtaining authenticity guarantees.

4.4 Security Requirements (Fundamental Architecture Requirements)

To be adoptable in a real operational context, the proposed architecture must not only solve interoperability issues (as discussed in Section 4.2), but must also satisfy a rigorous set of security requirements. These requirements are not generic, but derive directly from the constraints of the maritime domain.

- **Authenticity and Integrity**: These are the primary security requirements. The architecture must provide an irrefutable cryptographic guarantee that a message originates from the declared entity (sender authenticity) and that its content has not been altered during transmission (data integrity). This is the fundamental mechanism to mitigate spoofing attacks, which represent AIS's most critical vulnerability [6].

- **Efficiency and Low Overhead**: Given the nature of VHF channels, characterised by limited bandwidth and rigid packet formats (as analysed in Section 4.1), cryptographic mechanisms must not introduce excessive overhead. Efficiency is a binding requirement: solutions such as complete PKI digital signatures for every packet, although robust, are impractical in this context due to their size [2].

- **Scalability (Broadcast One-to-Many)**: AIS/VDES communications are inherently broadcast (one-to-many). The security solution must scale efficiently in high-traffic density scenarios (e.g., straits, port areas), where hundreds of receivers must authenticate messages from a single sender. The computational load on the sender and channel overhead must not grow linearly with the number of receivers [9].

- **Resilience (Fault and Attack Tolerance)**:

  1. **Packet loss tolerance**: VHF radio channels are inherently unreliable. The security protocol must be robust to packet loss and out-of-sequence reception, ensuring that authentication is still possible (e.g., by decoupling data from its authentication packet).
  2. **Attack resilience**: The solution must defend against specific attacks such as replay attacks (reinjection of valid but obsolete messages) and provide a security path against future threats, such as quantum attacks, favouring symmetric cryptographic primitives (as discussed in Section 2.3) [16].

- **Security Interoperability**: The interoperability requirement (discussed in Section 4.2) extends to security itself. The chain of trust must not be interrupted in the transition between the IP domain (SECOM) and the non-IP domain (VDES). The security guarantees acquired in the IP world must be preserved or mapped in a cryptographically valid manner to the VHF channel [6].

4.5 Definition of Use Case Scenarios

To validate the effectiveness and robustness of the proposed architecture, its implementation must be tested against use case scenarios that represent the critical communication flows of the maritime domain. The selected scenarios are not exhaustive but were chosen as they expose the system to the fundamental challenges identified previously, particularly IP/non-IP interoperability (Section 4.2), legacy constraint management (Section 4.3) and broadcast scalability (Section 4.4).

- **Ship-to-Port (Ship $\leftrightarrow$ Port)**: This is the primary scenario for testing the security gap in the "last mile" via VHF. Here, an IP-based entity (the port, operating within the SECOM ecosystem) receives messages from a non-IP entity (the ship, transmitting via VDES). This scenario is essential for:

  1. Validating the effectiveness of TESLA authentication in messages exchanged through VDES (including ASM/AIS services) [4].
  2. Testing the resilience of TESLA authentication to packet loss over the VDES radio channel.
  3. Demonstrating that security is added (via the parallel VDE-TER channel for TESLA data) without compromising the reception of standard VDES/AIS messages by nearby non-updated legacy ships.

- **Port-to-Authority (Port $\rightarrow$ Authority)**: This scenario tests the "upward flow" of information from the periphery (radio channel) to the core (IP network). A port acts as an aggregator: it receives authenticated data from ships via VDES/TESLA and must forward it to a national authority (e.g., VTS, Coast Guard) via the secure IP network (SECOM). The objective is to demonstrate that the authenticity of the data from the ship, verified by the port, is maintained and routed cryptographically in subsequent IP forwarding, ensuring an end-to-end chain of trust [6].

- **Authority-to-Authority (Authority $\leftrightarrow$ Authority)**: This scenario represents high-level data exchange, typically within the IP ecosystem of the Maritime Connectivity Platform (MCP). Although it may not directly involve the VHF channel, it is crucial for overall integration validation. The objective is to demonstrate that data originating from a ship (authenticated with TESLA) and forwarded by a port (Scenario 2) can be exchanged between different authorities using standard SECOM protocols, without any loss of security consistency and identity [7].

The joint analysis of these three communication flows allows the evaluation of the architecture not as a set of isolated components but as an integrated system capable of ensuring operational continuity and interoperability of security across the technological boundaries of the maritime domain.

Chapter 5

Proposed Architecture

5.1 Overall Vision: VDES and SECOM Interoperability

The proposed architecture addresses the authenticity requirement for AIS/ASM transmissions, without violating backward compatibility constraints, by introducing an "out-of-band" authentication model. The objective is not to modify the legacy AIS flows but to enrich them with a parallel security layer.

The architecture does not map SECOM payloads but uses them as a supporting infrastructure for a TESLA protocol focused on VHF messages [8]. The solution is based on three principles:

- **Channel Decoupling**: AIS/ASM messages continue to be transmitted on their designated VHF channels, ensuring compatibility with all legacy receivers. In parallel, the VDE-TER channel is used exclusively to transport TESLA authentication metadata. [4].
- **Trust Anchor via MCP/MIR**: The TESLA protocol, which is based on symmetric cryptography, requires a secure initial exchange of key chain commitment (K_0) [8]. The architecture leverages the SECOM infrastructure and the Maritime Identity Registry (MIR) precisely for this function: the MIR acts as a PKI trust anchor to distribute and certify TESLA commitments, securely binding a digital identity (MCP) to the operational identity (the TESLA key chain) of a ship or station [7, 6].
- **Authentication Layer**: Instead of an SECOM to VDES mapping layer, the architecture implements an authentication generation and validation layer. This software component runs in parallel to the AIS transponder to generate and verify TESLA data associated with transmitted AIS/ASM messages.

5.2 Use of VDE-TER as Parallel Channel for TESLA Authentication

To overcome the structural constraints of the AIS/ASM channels, which prevent the integration of "in-band" cryptographic data, the architecture adopts an "out-of-band" approach. The VDE-TER channel is used as a high-capacity parallel vector, dedicated exclusively to transport authentication metadata [4].

The operation decouples the data flow from the security flow:

- **Legacy Flow (AIS/ASM Channels)**: AIS and ASM messages continue to be transmitted according to the ITU standard, using their dedicated TDMA slots on VHF [3]. This ensures total backward compatibility, as non-VDES legacy receivers undergo no operational modification.
- **Authentication Flow (VDE-TER Channel)**: In parallel, the sender's authentication layer generates and transmits TESLA protocol messages on the VDE-TER channel. These include the MAC Payload and, after the predefined security delay, the Key Payload (containing the disclosure key of the MAC packets).

On the receiver side, a VDES-enabled unit acquires both flows. The validation software correlates the two flows, buffering AIS/ASM messages until reception of the corresponding Key Payload, which enables cryptographic verification.

This modular approach offers decisive advantages: it shifts the entire cryptographic overhead to the VDE-TER channel, leaving the legacy VHF channel unchanged. Furthermore, it exploits TESLA's broadcast efficiency (one-to-many), where a single MAC Payload and a single Key Payload per interval are sufficient to authenticate messages to an unlimited number of receivers [9].

5.3 Trust Anchor: Integration with MCP and Maritime Identity Registry (MIR)

The security of the TESLA protocol is based on authenticated knowledge of the initial commitment to the key chain (K_0) [8]. Being a symmetric key mechanism, TESLA cannot autonomously establish an initial trust relationship.

To fill this gap, the architecture anchors the TESLA bootstrap phase to an existing public key infrastructure (PKI) provided by the Maritime Connectivity Platform (MCP) and the Maritime Identity Registry (MIR) [7].

In this model, MCP and MIR do not intervene in the operational data flow but serve as a strategic trust anchor for the trust initialisation phase:

- **Identity Validation**: The MIR provides the standard mechanism for digital identity validation of a maritime entity (ship, port authority, control centre, etc.) through X.509 certificates issued within the MCP PKI [13]. This allows a certified public key to be uniquely associated with a verified identity.

- **Secure Distribution of TESLA Commitments**: The architecture leverages the MIR as a public and reliable registry for publishing the TESLA commitments (root keys). Each entity (e.g., a ship) that intends to send authenticated messages generates its own key chain $\{K_n, \ldots, K_0\}$ and submits the commitment K_0, signing it with its private key associated with the MCP certificate. Any receiver can query the registry, obtain the sender's public key (via the MCP certificate), verify the commitment's signature to securely retrieve K_0, and thereby establish a verifiable cryptographic link between the certified identity (MCP) and the symmetric operational chain used by TESLA [7].

- **Fallback Management and PKI Interoperability**: Although high-frequency operational authentication is efficiently managed by the TESLA protocol, the underlying PKI remains available to guarantee non-repudiation functionality or as a fallback mechanism in case of protocol failure. In these scenarios, standard SECOM mechanisms based on digital certificates and secure TLS exchange can be used [6].

- **Quantum Security Considerations**: Current MCP/MIR PKI implementations are based on classical algorithms (RSA or ECDSA) that are not resistant to quantum attacks [16]. However, the proposed architecture leverages the fact that TESLA is a symmetric authentication protocol, and the only asymmetric component (thus the quantum vulnerability) is the digital signature used to certify the Root Key (K_0) or Commitment.Consequently, the TESLA authentication system is inherently adaptable: its resistance to quantum attacks is directly inherited from the security level of the PKI managing the commitment. To achieve quantum-proof security, the strategy is two-fold:

  1. **PKI Migration**: It is necessary to migrate the MCP/MIR PKI infrastructure to standardised post-quantum algorithms (such as CRYSTALS-Dilithium, Falcon, SPHINCS+, or Kyber). As soon as the certificates are updated and used to sign the TESLA Commitments, the entire authentication system automatically acquires quantum resistance.

  2. **Alternative Management System**: Alternatively, it is possible to design an independent certificate management system dedicated exclusively to publishing and validating TESLA commitments, based on cryptographic primitives that are already inherently quantum-resilient [17].

5.4 Communication Workflow and Authentication Processes

The complete operational flow of the proposed architecture is articulated in three distinct phases:

1. **Setup and Synchronisation Phase**: Entities (ships, stations) execute the bootstrap protocol described in Section 6.2.2. This process, which uses the VDE-TER channel, allows participants to synchronise their clocks (a fundamental requirement of TESLA [8]) and exchange commitments. Trust in this exchange is guaranteed by preliminary validation of identities and commitments through the MCP/MIR infrastructure [7].

2. **Operational Transmission Phase**: During navigation, the process is asynchronous and parallel:

   - The ship transmits its standard AIS/ASM messages on legacy VHF channels.
   - Simultaneously, the ship's authentication layer calculates the MAC for these messages and transmits the corresponding MAC payload on the VDE-TER channel.
   - After the predefined security delay (e.g., intervals), the layer transmits the key payload (disclosure keys) also via VDE-TER.

3. **Asynchronous Validation Phase**: VDES-enabled receivers acquire both flows:

   - AIS/ASM messages are received and buffered as "unverified".
   - TESLA packets received via VDE-TER are processed by the validation layer.
   - Upon arrival of the pertinent Key Payload, the validator verifies its authenticity and, if valid, uses it to verify the MACs of the buffered AIS/ASM messages. Messages that pass the check are promoted to "authenticated" status.

Chapter 6

Design and Adaptation of the TESLA Protocol

The TESLA (Timed Efficient Stream Loss-tolerant Authentication) protocol is a solution designed to provide efficient authentication in broadcast and multicast scenarios. Its architecture solves the fundamental dilemma of broadcast authentication: the inefficiency of asymmetric cryptography (digital signatures per packet) and the insecurity of symmetric cryptography (a shared MAC, which would allow any receiver to impersonate the sender) [9].

TESLA solves this problem by introducing time-based asymmetry, using almost exclusively symmetric cryptographic primitives (hash functions and MACs), which are orders of magnitude more efficient than digital signatures.

6.1 The Original TESLA Protocol: Fundamental Principles

The original design of TESLA, as described in RFC 4082, is based on three interdependent conceptual pillars: loose time synchronisation, a one-way key chain, and delayed key disclosure [8].

6.1.1 Basic Requirements: Loose Time Synchronisation

An essential requirement for TESLA operation is the loose time synchronisation between the sender and all receivers. Unlike protocols that require tight synchrony, TESLA does not require perfectly aligned clocks.

The requirement, as defined in RFC 4082, is that each receiver knows an upper bound (D_t) for the lag of its clock relative to the sender's clock. In formal terms, if the receiver's local time is T_r and the sender's time is T_s, the receiver must be able to guarantee $T_s < T_r + D_t$. This upper bound D_t is typically established during an initial bootstrap phase, for example, through a simple two-way handshake [8].

This synchronisation is fundamental because it allows the receiver to perform a “safe packet test”: a receiver can reliably verify whether a received packet is “safe,” i.e., whether the cryptographic key used to authenticate it should not yet have been disclosed by the sender.

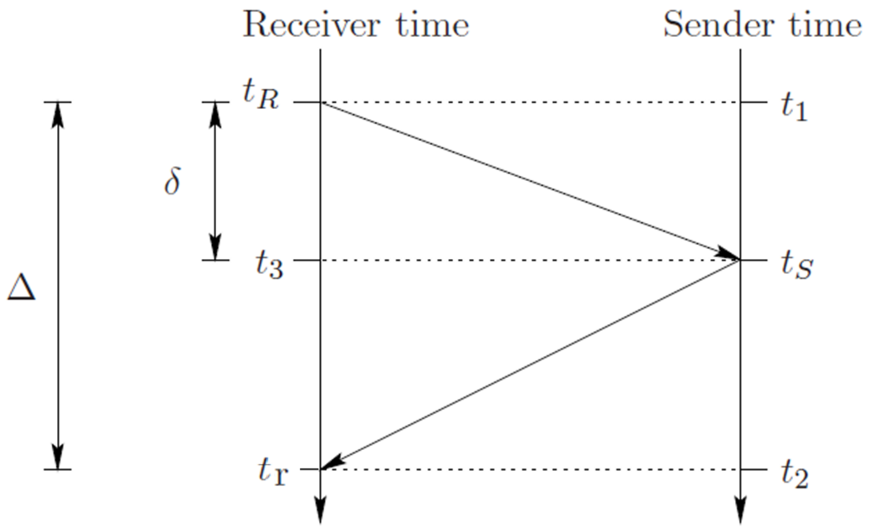


Figure 6.1: TESLA loose synchronisation

6.1.2 Central Mechanism: One-Way Key Chain

The central cryptographic mechanism of TESLA is the one-way key chain. This is a sequence of secret keys generated such that, given a key K_i, it is easy to calculate the previous key K_{i-1}, but it is computationally intractable to derive the next key K_{i+1} [8].

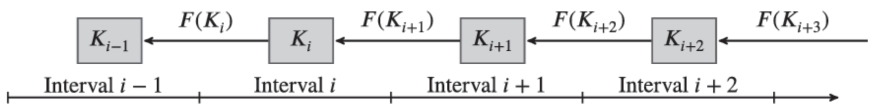


Figure 6.2: One-Way Key Chain generation

The generation process, described in RFC 4082, occurs backwards:

1. The sender chooses a random secret value K_N, which will be the last key in the chain.

2. Recursively applies a pseudo-random function (PRF) or a cryptographic hash function F (e.g., $F(k) = H(k)$) to generate the entire chain: $K_{i-1} = F(K_i)$.

3. This process is repeated N times until the key K_0 is reached.

The final value K_0 is called the "commitment" (or "root" of the chain). This commitment is the only part of the chain that is made public and securely distributed to all receivers during bootstrap. It acts as a trust anchor: any receiver that has authenticated K_0 can verify the authenticity of any subsequent key K_i by applying the function F for i times and comparing the result with K_0.

6.1.3 Security Guarantee: Delayed Key Disclosure

Delayed disclosure is the fundamental mechanism of TESLA that couples loose time synchronisation with a one-way key chain to create computational asymmetry [9].

The asymmetry arises because, at the moment a packet from interval i is received, the corresponding key k_i is still secret, preventing immediate authentication. Only in the next interval, when k_i is disclosed (e.g., inside packet P_{i+1}), the receiver can authenticate the content associated with interval i. This ensures that messages can be verified only after disclosure, that an attacker cannot forge valid MACs in real-time, and that the receiver can continuously authenticate past messages while buffering new ones. The core authentication logic embedded in each transmitted packet proceeds as follows:

Packet Generation (Sender)

In interval i, the sender generates the authentication code for the current message M_i using the current undisclosed key k_i:

$$\text{MAC}_i = \text{HMAC}_{k_i}(M_i)$$

The packet sent in the interval i contains:

- The message M_i.
- The corresponding MAC MAC_i.
- The disclosure key k_{i-1}, i.e., the key of the previous interval (or the key corresponding to the required security delay d).

This matches the logical structure shown in Figure 6.3:

$$P_i = \{\, M_i \,\|\, \text{MAC}_i \,\|\, k_{i-d} \,\}$$

where $d = 1$ represents the minimal delay.

Packet Verification (Receiver)

Upon receiving packet P_i, the receiver performs two distinct operations:

1. **Safe-Receive Test**:

   The receiver checks that the key used to compute MAC_i (the still-undisclosed k_i) has not yet appeared in any received packet. If this condition holds, the pair (M_i, MAC_i) is buffered [8].

2. **Authentication of Buffered Messages**:

   The disclosed key k_i is validated by checking that applying the one-way function yields the previously known commitment (e.g., $F(k_i) = k_{i-1}$ or ultimately $F^i(k_i) = k_0$). Once validated, k_i is used to authenticate all buffered messages that were generated in interval i.

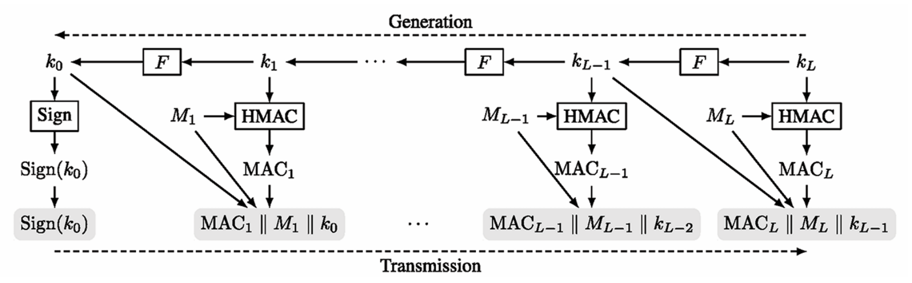


Figure 6.3: TESLA Authentication Algorithm

6.1.4 Limitations of Direct Application to Maritime Broadcast

Although robust, the original TESLA protocol (RFC 4082) presents three complexities when directly applied to the VDES domain:

1. **Bootstrap Management**: The protocol assumes the existence of a secure channel to distribute the commitment K_0 and to perform time synchronisation. In a global and decentralised system such as the maritime one, establishing this secure channel for each ship ("late joiner") represents a significant infrastructural challenge [2].

2. **Message-Key association**: A traditional "in-band" security model, which binds cryptographic metadata (such as MACs and keys) to each individual message, is highly inefficient in scenarios characterised by continuous data flows and packets with reduced payload (e.g., AIS). In this context, the cryptographic overhead per packet consumes a disproportionate fraction of the channel capacity, which is already inherently limited. The need thus emerges for an architecture that decouples the transport of authentication metadata (MACs and keys) from the data payload.

3. **Variable Latency and RTT**: The TESLA protocol relies on a strictly defined time delay (δ) (described in Section 6.1.1) between the reception of a message

authenticated with a key K_i and the disclosure of the key K_i. This time δ must be greater than the expected maximum Round-Trip Time (RTT), plus any time synchronisation error, to prevent immediate use of the key for forgery. However, in the maritime environment, the RTT between a coastal station (VTS/shore) and a vessel varies significantly based on the ship's constantly changing geographical position, the radio propagation conditions, and the specific VDES communication link being utilised (e.g., satellite vs. terrestrial VHF). This variability makes it impossible to define a single, strict maximum RTT suitable for all scenarios globally, forcing the choice of an overly large δ (reducing timeliness) or risking security violations in edge cases.

These limitations require a redesign of the protocol, specifically for the maritime domain, as discussed in the next section.

6.2 Redesign of TESLA for the Maritime VDES Domain

The redesigned TESLA architecture introduces several key deviations from the standard model defined in RFC 4082. These changes adapt the protocol to the operational constraints of the maritime VDES environment and to the characteristics of AIS traffic.

6.2.1 Architectural Deviations from RFC 4082

1. **Decoupling of Key Management from Message Flow**

   The standard TESLA often implies, or is implemented with, a tight coupling between cryptographic material and the message flow (e.g., associating a unique key or commitment index to every single message, or requiring frequent key changes rigidly linked to the smallest time granularity).

   In the redesigned model, fixed-duration time slots (T_{slot}) are used for key management:

   - Instead of having a key associated with every message, one key (K_i) is associated with one specific time slot (T_{slot}), which has a predefined duration (e.g., 3 seconds).
   - Multiple messages can be securely transmitted for the duration of the same T_{slot} using the same key K_i.
   - The key K_i used to authenticate messages within the slot is disclosed only after the entire T_{slot} has elapsed, ensuring the necessary time delay for receiver validation while maximising channel efficiency.

This shift replaces the rigid message-by-message key association with an authentication mechanism based on time-based batch processing, significantly reducing cryptographic overhead and improving efficiency in low-bandwidth, high-frequency data environments like VDES [4].

2. **Separation of Authentication Material into Independent Packets (Out-of-Band Authentication)**

   RFC 4082 assumes an *in-band* model, where each authenticated message must carry ($\mathbf{M} \parallel \mathbf{MAC} \parallel \mathbf{K_{i-d}}$) in a single packet.

   In the redesigned model:

   - Authentication information is split into three independent packet types (as detailed in Section 7.4): the **Data Packet** (which contains only the AIS payload), the **MAC Packet** (containing the MAC and a linking structure) and the **Key Packet** (containing only the disclosed key).

   This is a major architectural deviation, enabled by TESLA's temporal asymmetry, as authentication does not need to be bundled with the data as long as the temporal alignment is preserved.

3. **Keys Are Disclosed in Dedicated "Key Packets"**

   In standard TESLA, the disclosed key K_{i-d} travels inside every authenticated packet.

   In the redesigned model:

   - The disclosure keys are no longer carried in the same packet as the MAC.
   - Each key is sent in a dedicated key packet, drastically reducing the size of MAC packets and minimising bandwidth overhead.

4. **Modified Packet Generation (Sender)**

   To maintain cryptographic binding between the MAC and its corresponding data (now transmitted in separate frames), a 32-byte **Linking Structure** is introduced.

   In interval i, the sender first derives the linking structure L_i from the hash of the current message M_i:

$$L_i = \mathrm{H}(M_i)$$

   The sender then generates the authentication code for the linking structure using the current undisclosed key k_i:

$$\mathrm{MAC}_i = \mathrm{HMAC}_{k_i}(L_i)$$

   The packet sent in the interval i contains:

- The truncated to 8-bytes Linking Structure L_i (which links the MAC to the separated data M_i).
- The corresponding MAC MAC_i.

This results in the following logical structure:

$$P_i = \{\, L_i \,\|\, \mathrm{MAC}_i\}$$

This mechanism ensures cryptographic coupling even when data and MAC travel in separate frames, linking the contents regardless of their temporal separation.

5. **Adaptation to VDES Multichannel Architecture**

The redesign exploits the intrinsic VDES architecture:

- Data packets continue to be transmitted on AIS channels.
- MAC and key packets are transmitted on the higher-bandwidth VDE-TER channel.

This multimodal separation is not present in RFC 4082 but becomes a natural optimisation in the maritime environment.

6.2.2 Bootstrap Phase: Time Synchronisation and Commitment Distribution

To address loose time synchronisation and ensure the secure exchange of cryptographic commitments, such as the initial key commitment (K_0), the bootstrap protocol must operate over a trusted channel. This is achieved by implementing a "three-way handshake" that leverages an underlying Public Key Infrastructure (PKI), such as that provided by the Maritime Connectivity Platform (MCP) and the Maritime Identity Registry (MIR), to authenticate the bootstrap messages themselves. While MIR is the reference in the maritime domain, other X.509 certificate-based infrastructures can fulfil the same "trust anchor" function [7].

The process flow, as illustrated in Figure 6.4, is as follows:

1. Announcement (A $\rightarrow$ B)

Station A (the announcer, Source ID: STA1) periodically transmits a certified **Announcement** packet. This message serves as the first authenticated exchange the "I am ready to listen" signal and enables receiver bootstrapping.

- The packet content includes the station's identifier (name_A) and the current timestamp (t_{rA}).

- It is authenticated through a digital signature calculated using A's private PKI key and the corresponding MCP/MIR certificate [13].
- This initial signature allows any receiver (like Station B) to verify the sender's identity and establish trust before proceeding with the efficient, symmetric TESLA authentication.
- Upon receiving, Station B receives the announcement from A and stores t_{rA}.

2. Sync Request (B $\rightarrow$ A)

Subsequently, Station B (the Listener, Source ID: STB1) responds with a certified **Sync Request** packet. This message is critical for security and is authenticated (e.g., digitally signed) using B's cryptographic identity (obtained from the PKI infrastructure, e.g., MIR).

The packet contains several key parameters:

Identification and Temporal Alignment:

- **Identifiers:** For the sender (name_B) and the intended recipient (name_A).
- **Received Timestamp ($\mathbf{t}_{rA}$):** The timestamp from the Announcement message, which acts as proof of freshness to guarantee a direct, non-replayed link between the two exchanges.
- **B's Current Time ($\mathbf{t}_{sB}$):** Current time of Station B at the moment of transmission.
- **Key Chain Start Time (start_sending_time):** The beginning of the current keychain's operational period. This is the temporal reference point against which the current key index is aligned, reflecting the system's time slot structure.

TESLA Configuration and Cryptographic Commitment:

- **Commitment Key ($\mathbf{K_B}$):** B's initial commitment key, necessary for A to authenticate B's future messages.
- **Current Key Index:** The index of the key currently in use is directly correlated with the keychain start time to indicate the active time slot.
- **TESLA Configuration:** The operational TESLA parameters (e.g., hash chain length, base time interval, packet buffer size, and key disclosure delay) required for A to successfully decode and verify B's subsequent authenticated messages.

PKI Authentication Data:

- The complete certificate and the digital signature of the payload message, ensuring its authenticity and integrity.

Upon reception, Station A verifies authenticity and integrity through the certificate and signature. If valid, A securely stores the trusted commitment $\mathbf{K_B}$ and the start_sending_time, configures its receiver based on the TESLA parameters provided, and calculates the necessary time offset for synchronisation.

3. Sync Response (A $\rightarrow$ B)

Finally, Station A sends a certified **Sync Response** packet. This confirms synchronisation and functions as an acknowledgement message (ACK). This message is authenticated using A's cryptographic identity (from the PKI infrastructure) and contains:

- **Identification:** Identifiers for the sender (name$_A$) and the receiver (name$_B$).
- **B's Timestamp (t$_{sA}$):** Station B's time sent in the Sync Request message.
- **PKI Authentication Data:** The full certificate and the digital signature of the message payload, guaranteeing its authenticity and integrity.

Upon reception, Station B verifies the digital signature using A's certificate. Station A, instead, confirms that the synchronisation is complete and calculates the Round-Trip Time (RTT) to finalise the temporal alignment.

Bootstrap Completion

At the end of this authenticated bootstrap, A has synchronised its clock with B, and the cryptographic commitment $\mathbf{K_B}$ is securely established from B to A. The reliability of the entire subsequent TESLA authentication chain is fundamentally based on the PKI authentication of this initial three-way exchange.

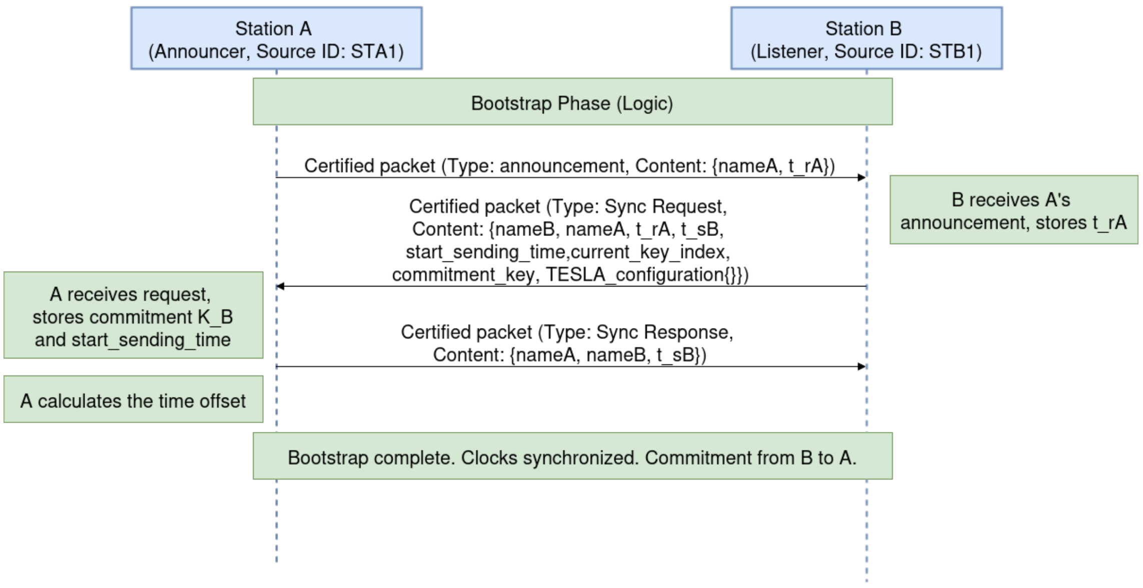

Station A
(Announcer, Source ID: STA1)
Station B
(Listener, Source ID: STB1)
Bootstrap Phase (Logic)
Certified packet (Type: announcement, Content: {nameA, t_rA})
B receives A's announcement, stores t_rA
Certified packet (Type: Sync Request, Content: {nameB, nameA, t_rA, t_sB, start_sending_time,current_key_index, commitment_key, TESLA_configuration{}})
A receives request, stores commitment K_B and start_sending_time
Certified packet (Type: Sync Response, Content: {nameA, nameB, t_sB})
A calculates the time offset
Bootstrap complete. Clocks synchronized. Commitment from B to A.

Figure 6.4: Bootstrap phase: Three-way handshake for PKI-authenticated TESLA commitment and time synchronisation.

6.2.3 Key Generation, Management, and Disclosure for Time Intervals

The cryptographic key lifecycle management mechanism is the operational core of the redesigned TESLA architecture. It is implemented through a key chain component that guarantees authenticity and integrity over time while ensuring controlled key rotation and computational efficiency. The process is articulated in distinct phases that ensure cryptographic continuity between successive chains.

Key Chain Generation

The keychain generation process is based on an iterative hash operation. Starting from a secret seed, which serves as K_n (the top of the chain), a secure hash function H (e.g., SHA-256) is applied to generate the one-way sequence backward, according to the relation:

$$K_i = H(K_{i+1})$$

The entire chain $[K_0, K_1, \ldots, K_{n-1}]$ is generated and stored in operational order (from first to last use). The value K_0 (generated last) serves as the commitment: a trust anchor that is made public to allow a posteriori verification, while the sequence $K_1 \ldots K_n$ remains secret [8].

Initialization and Cryptographic Commitment Phase

The bootstrap phase of the system (detailed in Section 6.2.2) is dedicated to securely initiating a new key chain and distributing the initial cryptographic reference. The

Manager (the temporal and logical control unit) initiates the generation of a One-Way Key Chain $\{K_N, \ldots, K_0\}$ starting from a cryptographic seed. The output of this generation process is the commitment, which is the root element of the chain and the full key chain. As illustrated in the bootstrap flow diagram, the manager's crucial task is to certify and disseminate this commitment to the verifier (the receiver). This dissemination establishes the initial cryptographic reference, enabling the verifier to authenticate all subsequently released keys.

Usage and Disclosure Phase (Chain Consumption)

The operational phase unfolds over discrete, sequential time intervals (time slots), regulated by the manager. This process ensures efficient authentication of messages and the adherence to the fundamental security constraint of TESLA's. At each advance to the time interval (i):

1. **Message Authentication (Sender):** The system provides the sender (the entity that applies the MAC) with the current key K_i. This key is used to generate the Message Authentication Code (MAC) associated with the transmitted data.
2. **Delayed Disclosure (Verifier):** During the same time, the system provides the verifier with the key used in a previous time interval, K_{i-d}, where d represents the predefined security delay (disclosure delay).

This simultaneous state mechanism is vital for the protocol's security: it ensures that the keys used to authenticate a message are only disclosed after the validity interval of that message has expired. This prevents an attacker from forging a MAC, as the key needed to generate it is not yet available. The chain is considered exhausted when all keys have been used both for authentication and disclosed for verification.

Automatic Reload and Operational Continuity (Re-keying)

To ensure continuous service operation without the need to repeat the initial asymmetric (PKI-based) bootstrap phase, an automatic key rotation (re-keying) mechanism is implemented. This mechanism is crucial because it overcomes the inherent limitation of TESLA protocols, where the maximum number of keys must be fixed at the chain's initialisation.

1. **Transition and New Chain Generation**: At the end of a chain cycle (e.g., after using the last key K_N), the Manager activates the Refill procedure. This procedure generates a new One-Way Key Chain (e.g., $\{K'_M, \ldots, K'_0\}$) from a new cryptographic seed.
2. **Anchoring the New Commitment**: To ensure a cryptographically secure transition and maintain trust, the new commitment (K'_0) is not disclosed via a

new PKI bootstrap. Instead, it is transmitted as an ordinary message but authenticated using the TESLA mechanism of the preceding chain. The message containing the key K'_0 is thus made available in plaintext, but its authenticity is guaranteed by a MAC generated with one of the last still-active keys of the original chain (e.g., K_N).

3. **Overlap and Trust Continuity**: This "chaining" of commitments ensures cryptographic trust continuity and, crucially, allows the system to operate indefinitely by linking successive chains. The system enters a controlled transition phase where the two chains coexist: the system continues to disclose the last keys of Chain 1 (for retroactive verification of older messages) while simultaneously beginning to use the first keys of Chain 2 to authenticate new messages.

This model realises a temporal key rotation in a distributed environment, ensuring that every authenticated message can be verified in a subsequent interval, that operational continuity is maintained, and that the system is not constrained by a predefined key budget.

6.2.4 Buffering Mechanism and Asynchronous Verification in the Receiver

The reception subsystem is designed to asynchronously manage the decoupled flow of data payload and authentication metadata (MAC payload, key payload). The process is based on intermediate storage (buffering) that allows reconciling data with their cryptographic proofs, which arrive at different times due to delayed disclosure.

Reception Phase and Asynchronous Buffering

The receiving station is configured to process the two distinct operational payloads transmitted by the sender on the “out-of-band” channel (VDE-TER):

1. **MAC Payload**: Containing the MAC, the key index (i) of the signing interval, and the linking structure (the identifier, e.g., hash, that links it to the Data Payload).
2. **Key Payload**: Containing the disclosure key K_i and its index (i).

To handle non-deterministic packet arrival and potential packet loss over the radio channel (VHF), the receiver implements two logical buffer areas, utilising the linking structure as the correlation key to associate the raw data with its Authentication Code (MAC):

- **Data Buffer**: Stores the received data, indexed by its linking structure. Crucially, for every received data element, the buffer also records the presumed sending time slot (or key index) of every transmitting peer. This is necessary because in this implementation the data itself is treated as a black box, and the source cannot be certainly identified from the data payload alone. Storing the key index per peer allows the system to retrospectively attribute the data to the correct sender during the asynchronous verification phase.

Table 6.1: Example of Data Buffer Contents During TESLA Asynchronous Verification

Linking Structure	**Payload**	**Presumed Peer Indices**
`0x0FEE2B`	DATA	A:102 B:107 C:100
`0xCF5747`	DATA	A:102 B:107 C:100
`0xA781C1`	DATA	A:102 B:106 C:99
`0x840462`	DATA	A:101 B:106 C:99
`0x52BA36`	DATA	A:101 B:105 C:98

- **MAC Buffer**: Stores the MAC Payload received, which includes the key index of the temporal interval to which it belongs, the MAC itself, and the linking structure of the message that needs to be authenticated.

Table 6.2: Example of MAC Buffer Contents for one Peer

Key Index	**MAC Payload**	**Linking Structure**
107	`7F19A4...`	`0xCF5747`
106	`33A8D1...`	`0xA781C1`
106	`D99271...`	`0x840462`

This dual buffering structure, supported by the linking structure and associated temporal information, is essential to enable the asynchronous verification of data, as the MAC and the key required for verification arrive at different times.

Asynchronous Verification Phase

The verification mechanism, triggered by the reception of the key K_i, develops according to the following logical sequence:

1. **Verification of the revealed key (Chain Validation)**: The authenticity of the just-received key K_i is checked, ensuring it belongs to the known trust

chain (commitment K_0 or the previously validated key K_{i-1}), verifying the relationship $H(K_i) = K_{i-1}$. In case of failure, the key is discarded.

2. **Key-MAC Association (Key-to-MAC Lookup)**: For the validated key K_i, the system searches the MAC buffer for all MAC payloads with key index $= i$ (that is equal to the reception time interval).

3. **MAC-Data correlation (MAC-to-Data Lookup)**: For each MAC payload found, the linking structure and the key index are used to locate the corresponding data payload in the data buffer. If the data are not present (the message was never received or was lost), the MAC is discarded.

4. **Final Cryptographic Verification (MAC Validation)**: If both components (Data and MAC) are available, the system calculates MAC_{K_i}(Data payload) and compares it with the received and buffered MAC.

    - **Success**: If the comparison is positive, the message is marked as verified, and the payload is made available to the upper application layers.
    - **Failure**: Otherwise, the MAC is discarded as it is considered corrupted or inauthentic.

This asynchronous approach, based on buffering and delayed verification, allows the receiver to operate in non-deterministic network conditions while maintaining authenticity guarantees.

Buffer Cleanup and Expiration

To maintain operational efficiency and mitigate Denial-of-Service (DoS) attacks, where an adversary attempts to fill the buffer with spurious packets, the receiver implements an expiration mechanism for both the Data Buffer and the MAC Buffer.

- **MAC Expiration:** MAC payloads are discarded after a predefined number of time intervals have passed without receiving the corresponding valid disclosure key (K_{i-d}). This threshold is strictly enforced because, past this point, verifying the MAC becomes cryptographically impossible. Furthermore, authenticating an outdated message is generally operationally irrelevant.

- **Data Expiration:** Data messages are also subject to automatic expiration. A data entry is purged from the data buffer when its operational relevance has expired for all potential transmitting peers. This is determined by analysing two factors:

    1. The list of probable key indexes (time slots) associated with the data entry for every potential peer (recorded during initial buffering).
    2. The maximum packet validity time parameter is established during the synchronisation phase for each peer.

If the current time exceeds the validity period defined by the key index of all potential sources, the data message is definitively considered invalid for authentication purposes and is discarded. This ensures that only relevant data that await verification remain in the buffer.

6.3 Security Properties Analysis of the Proposed Architecture

The TESLA architecture redesigned for VDES not only solves efficiency and compatibility problems but also guarantees fundamental security properties, essential for adoption in a critical environment such as maritime.

6.3.1 Robustness to Packet Loss and Resilience

Protocol resilience derives directly from the "out-of-band" architecture and the nature of the one-way key chain.

- **Data/MAC Loss Tolerance**: In a VHF radio channel, packet loss is a common event. Thanks to decoupling (described in Section 6.2.1), loss of a data payload or MAC payload is not catastrophic. If the data payload is lost, the corresponding MAC payload (identified by the linking structure) will be discarded at verification time, but this does not affect any other message. If the MAC payload is lost, the corresponding data payload cannot be authenticated, but the integrity of the rest of the communication is preserved.
- **Key Loss Tolerance**: The most significant resilience is in key payload loss management. If a receiver loses the packet containing the key K_i, it does not need to discard the buffered messages for the interval i. It is sufficient to wait for the reception of any subsequent key (e.g., K_{i+1} or K_{i+2}). Since the chain is one-way ($K_i = H(K_{i+1})$), the receiver can retroactively recalculate the key K_i from K_{i+1}, unlocking and authenticating all buffered messages. This property makes the protocol exceptionally robust to packet loss in the authentication channel.

6.3.2 Scalability in Broadcast Scenarios (One-to-Many)

TESLA's efficiency in broadcast scenarios (one-to-many) is one of its main advantages over PKI-based approaches (digital signatures). Scalability is nearly perfect, as the sender's workload is constant ($O(1)$) and independent of the number of receivers (N) [9].

- **Sender Efficiency**: The sender performs one MAC calculation per message and sends a single MAC payload and a single key payload (per interval) to the

VDE-TER multicast address. This single transmission is received simultaneously by N receivers (whether $N = 1$ or $N = 10{,}000$). The computational and bandwidth cost for the sender does not increase as the receivers increase.

- **Receiver Efficiency**: The computational load on the receiver side is extremely low. For each key, the receiver performs a single hash calculation (to verify the key if the previous key has already been verified) and one HMAC calculation for each buffered message. These are symmetric operations, orders of magnitude faster than verifying a digital signature (e.g., ECDSA), which requires complex operations on elliptic curves.

6.3.3 Post-Quantum Security (Quantum-Safety)

A long-term strategic advantage of this architecture is its intrinsic post-quantum security (quantum safety).

The entire security of the operational phase of TESLA (after bootstrap) is based exclusively on two symmetric cryptographic primitives:

1. Secure hash functions (e.g., SHA-256) for one-way chain generation.
2. Message Authentication Code (e.g., HMAC-SHA256) for packet authentication.

These symmetric cryptographic primitives are widely considered robust against known quantum computer attacks. No quantum algorithms are capable of fundamentally compromising the security of hash functions or MACs [17]. Although some theoretical attacks (such as Grover's algorithm) may reduce their effective strength, this simply requires the use of keys and hashes of adequate length (e.g., 256 bits) without invalidating the security paradigm [16].

Although the bootstrap phase (described in Section 6.2.2), which relies on the Maritime Identity Registry (MIR), uses traditional PKI and is therefore vulnerable to quantum attacks, the high-frequency operational phase offers better protection. Specifically, the authentication mechanism for AIS/ASM messages ensures quantum-safe properties for data flow integrity, thanks to the use of one-way hash chains and delayed disclosure techniques.

To ensure completely quantum-resilient end-to-end security, however, it will be necessary to adopt recognised post-quantum algorithms (e.g., CRYSTALS-Dilithium, Falcon, SPHINCS+, Kyber) in the MIR's PKI infrastructure or implement a dedicated and independent certificate management system based on cryptographic primitives resistant to quantum computing [6].

Chapter 7

Testbed Implementation and Prototyping

The analysis and design of the "out-of-band" architecture (described in Chapter 6) require practical validation to demonstrate its feasibility. This chapter describes the implementation of the software prototype built to emulate the VDES/TESLA communication flow and, in particular, to interface with real maritime transmission equipment [4].

7.1 Prototype Architecture and Technology Stack

The prototype was developed in Python, chosen for its rapid development capabilities and robust native libraries for low-level network manipulation and binary serialisation.

The core of this prototype is the "AUTH INTERCEPTOR SYSTEM", which functions as a transparent middleware positioned between the VDES Communication Module and the User Applications or the AIS. This interceptor manages the authentication workflow and is divided into two fundamental components:

7.1.1 Receiver Interface (VDES $\rightarrow$ Applications)

This module handles incoming traffic from the VDES channel (incoming AIS or Authentication message):

1. **Packet Sniffer:** Captures all incoming packets from the VDES Communication Module.

2. **Message Classifier:** Separate the captured packets into two streams: standard AIS messages (routed to the AIS Buffer) and Authentication messages (routed to the Auth Processor).

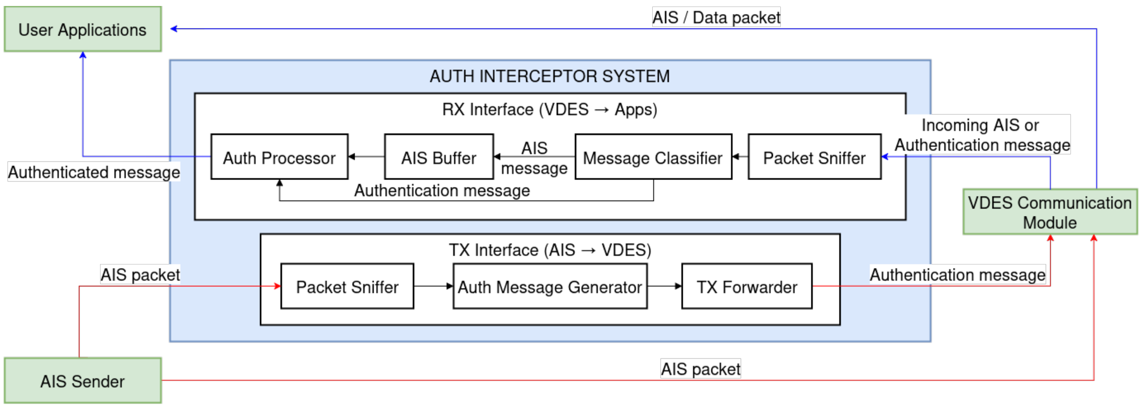


Figure 7.1: Auth interceptor system

3. **Auth Processor:** Receives the Authentication messages and uses the AIS Buffer (which implements the dual-buffer logic described in Section 6.2.4) containing the raw AIS data. The Auth Processor performs the asynchronous TESLA verification and, upon successful authentication, forwards the Authenticated message to the User Applications [8].

7.1.2 Sender Interface (Applications or AIS → VDES)

This module processes outgoing AIS data for transmission:

1. **Packet Sniffer:** Captures the AIS packets that require authentication.

2. **Auth Message Generator:** Implements the TESLA generation logic, applying the current key (K_i) to the captured AIS message to calculate the MAC, thus creating an Authentication message.

3. **Sender Forwarder:** Routes the newly created Authentication message to the VDES Communication Module for transmission (via the Blue path on Figure 7.1).

Note that the standard AIS packets (red path on Figure 7.1), originating from the AIS Sender, are directed to the VDES Communication Module without being routed through the Interceptor. The presence of the parallel red path signifies that the Interceptor is designed not to interfere with the standard AIS messaging flow, thereby ensuring it operates strictly in an out-of-band mode for authentication purposes only.

The interface between these modules (and the external VDES equipment) does not occur through simple REST APIs or TCP but must strictly adhere to the IEC 61162-450 standard for onboard communication to ensure realistic compatibility [18].

7.2 Experimental Environment and Physical Layer Emulation

The empirical validation of the cryptographic architecture proposed in this thesis requires an experimental framework capable of transcending theoretical abstraction to address the stochastic nature of the maritime channel. However, given the nascent deployment status of the VDES standard (specifically the scarcity of Commercial Off-The-Shelf (COTS) transceivers supporting the VDE-TER terrestrial component) direct validation on native hardware remains currently inaccessible for open research [5].

To overcome this limitation while maintaining fidelity to the physical constraints of the target environment, this research establishes a hybrid **Hardware-in-the-Loop (HIL)** testbed. This platform leverages **LoRa (Long Range)** technology as a functional physical layer surrogate, chosen for its isomorphic characteristics regarding bandwidth limitations, latency, and packet loss profiles compared to maritime VHF communications [19].

This chapter details the engineering of the experimental apparatus, structured into three distinct abstraction layers: physical radio emulation using LoRa transceivers, service orchestration via Docker-based virtualization, and the development of custom software modules that bridge the gap between the IEC 61162-450 shipboard network and the emulated radio link.

7.2.1 Physical Layer Abstraction: Using LoRa as VDES Surrogate

To validate the proposed TESLA-based authentication protocol under realistic electromagnetic constraints, the testbed employs **LoRa (Long Range)** technology as a functional surrogate for the VDES physical layer. Unlike purely software-based simulations, this approach subjects the protocol to genuine stochastic channel behaviours, including multipath fading, interference, and non-deterministic packet loss.

Although VDES operates in the VHF maritime band (approximately 160 MHz) using modulations such as GMSK (for legacy AIS) or $\pi/4$-QPSK and 16-QAM (for VDE-TER), LoRa operates in the 868 MHz ISM band (EU868) using the Chirp Spread Spectrum (CSS). Despite these spectral differences, LoRa was selected on the basis of three key technical isomorphisms that make it an ideal stress-test environment for secure maritime communications [19, 20]:

1. Throughput and Latency Constraints (The "ShortFast" Configuration):

   Transceivers were configured using the Meshtastic firmware with the *"ShortFast"* preset. This configuration utilises a Spreading Factor of 7 (SF7) and a bandwidth of 250 kHz. This yields a data rate comparable to the legacy AIS/ASM components of VDES (approximately 9.6 kbps), avoiding the excessive Time-on-Air (ToA) associated with higher Spreading Factors (e.g., SF12).

This setup imposes strict efficiency requirements on the authentication overhead, forcing the TESLA protocol to optimise the packet size and minimise cryptographic expansion.

2. Shared Medium and Collision Dynamics:

   Similar to the TDMA (Time Division Multiple Access) structure of VDES, the LoRa testbed operates on a shared medium. By utilising Meshtastic's Channel Activity Detection (CAD) and CSMA-like collision avoidance, the testbed simulates the contention and "busy channel" scenarios typical of congested maritime port environments. This creates a realistic adversary for the time-synchronisation requirements of the TESLA protocol, as channel access latency directly impacts the validity of delayed key disclosure intervals.

3. Broadcast Orientation and Topology:

   The experimental topology consists of Heltec WiFi LoRa 32 V3 nodes (equipped with the Semtech SX1262 chipset and ESP32-S3 MCU). The nodes are configured in *"Client Mute"* mode. This crucial setting disables the mesh routing and packet relaying features of Meshtastic, forcing the nodes to act as pure endpoints. This isolates the authentication logic from the mesh routing overhead, ensuring that measured latencies and packet losses are solely attributable to the physical link and the cryptographic processing time, simulating a direct ship-to-ship or ship-to-shore VDES broadcast.

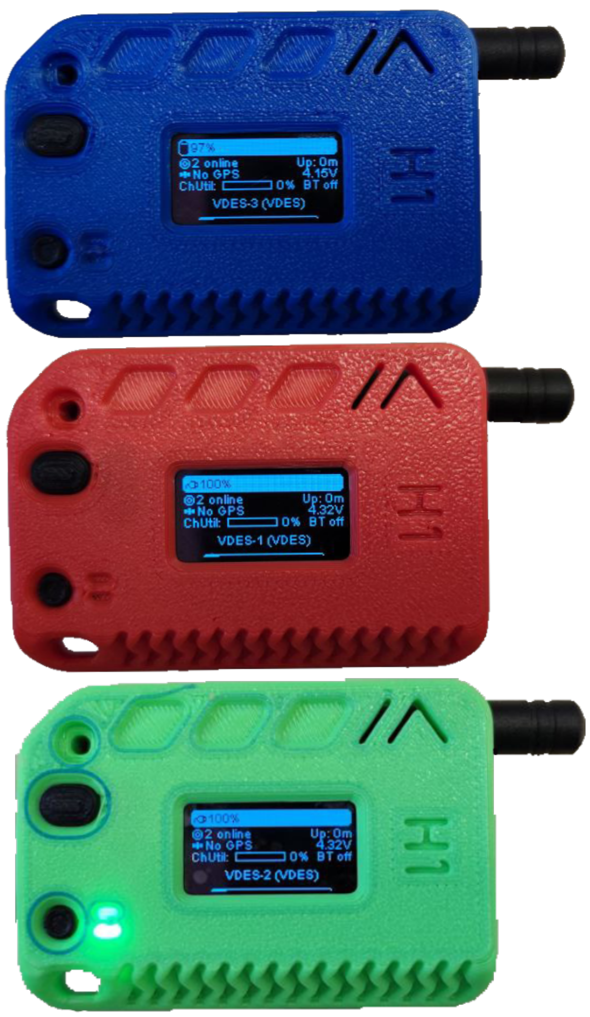


Figure 7.2: Heltec LoRa node antennas with Meshtastic firmware and 3D-printed case

By mapping the VDES frame constraints onto LoRa physical packets, this abstraction layer provides a conservative lower-bound evaluation environment: if the authentication protocol remains robust over the LoRa link, it offers strong assurances of its viability for higher-power VDES infrastructure.

7.2.2 Docker-Based Containerization and Orchestration

To ensure reproducibility, modularity, and strict separation of concerns, the entire software stack (excluding the embedded radio firmware) was engineered as a distributed system of microservices orchestrated via **Docker**. This architecture decouples the cryptographic protocol logic from the hardware interface, allowing for a precise simulation of the interaction between a ship's onboard systems and its communication equipment [21].

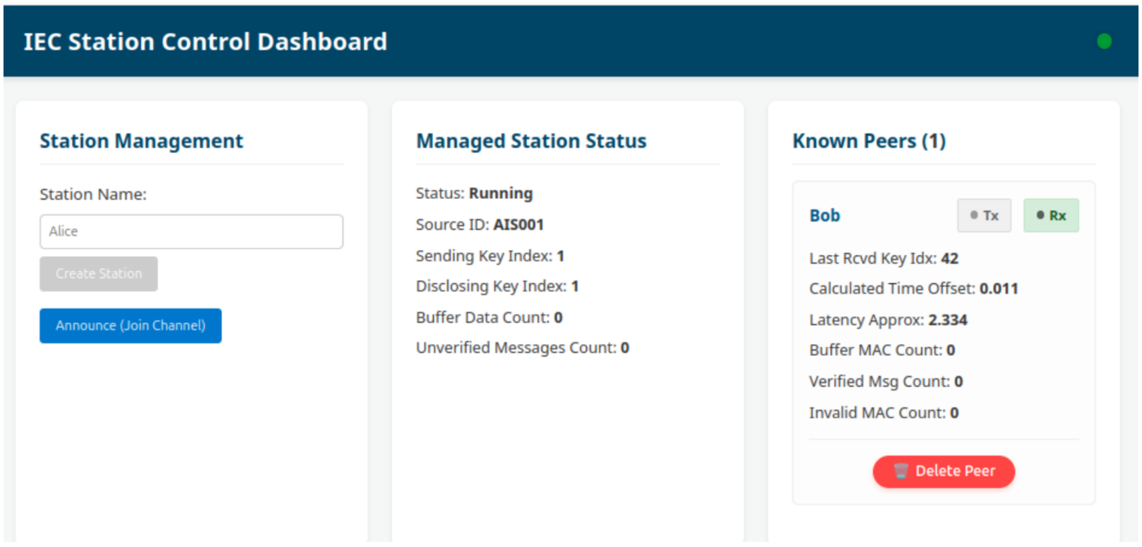
The orchestration is defined via a declarative `docker-compose` specification that establishes a virtualised network topology that mimics a real maritime environment.

The system comprises two primary services communicating over an isolated bridge network (`tesla-network`):

1. The Protocol Logic Service (`tesla`):

   This container encapsulates the core TESLA authentication implementation. Acting as the ship's secure communication controller, it generates cryptographic payloads and encapsulates them within **IEC 61162-450**-compliant UDP datagrams. Instead of directly interacting with hardware, this service transmits these packets over the virtual internal network, targeting the radio interface. This effectively simulates the flow of data to the VDES transceiver via the ship's Ethernet backbone.

   The operational status of this service is exposed via a web dashboard (Figure 7.3), which allows operators to manage the station identity (e.g., Alice) and monitor the synchronisation status of known peers (e.g., Bob).


IEC Station Control Dashboard

Station Management

Station Name: Alice

Managed Station Status

Status: **Running**
Source ID: **AIS001**
Sending Key Index: **1**
Disclosing Key Index: **1**
Buffer Data Count: **0**
Unverified Messages Count: **0**

Known Peers (1)

Bob

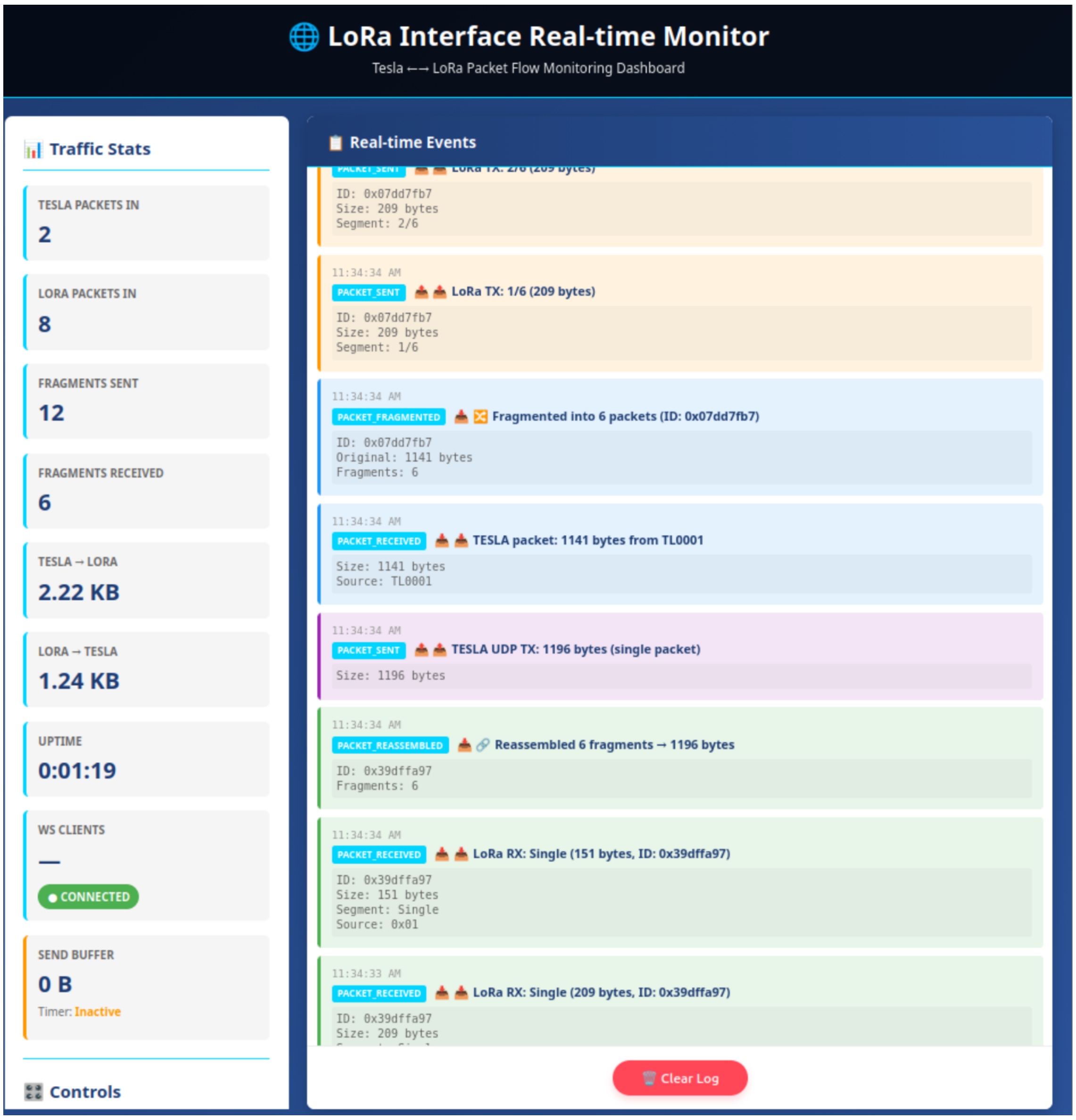
Last Rcvd Key Idx: **42**
Calculated Time Offset: **0.011**
Latency Approx: **2.334**
Buffer MAC Count: **0**
Verified Msg Count: **0**
Invalid MAC Count: **0**

Figure 7.3: IEC Station Control Dashboard. This interface visualises the state of the Protocol Logic Service, showing the local station parameters (Sending Key Index) and the real-time status of authenticated peers (Last Received Key Index and Buffer counts).

2. The Hardware Abstraction Layer (`lora-interface`):

   This service functions as a bridge between the virtual Docker network and the physical electromagnetic domain. Configured in `privileged` mode with direct device mapping (e.g., passing through `/dev/ttyUSB0`), it gains exclusive control over the serial connection to the Heltec LoRa nodes. Its primary role is to listen for incoming UDP packets from the `tesla` service, strip the IEC network headers, and serialise the payload for transmission over the LoRa physical link. Conversely, it captures incoming radio frames and forwards them back to the logic layer via UDP.

Figure 7.4 illustrates the real-time monitoring of this layer, highlighting the fragmentation process required to adapt the large TESLA UDP datagrams to the constrained LoRa physical frames.


LoRa Interface Real-time Monitor
Tesla ←→ LoRa Packet Flow Monitoring Dashboard
Traffic Stats
Real-time Events
Controls
Clear Log

```
TESLA PACKETS IN
2
LORA PACKETS IN
8
FRAGMENTS SENT
12
FRAGMENTS RECEIVED
6
TESLA → LORA
2.22 KB
LORA → TESLA
1.24 KB
UPTIME
0:01:19
WS CLIENTS
—
● CONNECTED
SEND BUFFER
0 B
Timer: Inactive
```

```
PACKET_SENT LoRa TX: 2/6 (209 bytes)
ID: 0x07dd7fb7
Size: 209 bytes
Segment: 2/6

11:34:34 AM
PACKET_SENT LoRa TX: 1/6 (209 bytes)
ID: 0x07dd7fb7
Size: 209 bytes
Segment: 1/6

11:34:34 AM
PACKET_FRAGMENTED Fragmented into 6 packets (ID: 0x07dd7fb7)
ID: 0x07dd7fb7
Original: 1141 bytes
Fragments: 6

11:34:34 AM
PACKET_RECEIVED TESLA packet: 1141 bytes from TL0001
Size: 1141 bytes
Source: TL0001

11:34:34 AM
PACKET_SENT TESLA UDP TX: 1196 bytes (single packet)
Size: 1196 bytes

11:34:34 AM
PACKET_REASSEMBLED Reassembled 6 fragments → 1196 bytes
ID: 0x39dffa97
Fragments: 6

11:34:34 AM
PACKET_RECEIVED LoRa RX: Single (151 bytes, ID: 0x39dffa97)
ID: 0x39dffa97
Size: 151 bytes
Segment: Single
Source: 0x01

11:34:33 AM
PACKET_RECEIVED LoRa RX: Single (209 bytes, ID: 0x39dffa97)
ID: 0x39dffa97
Size: 209 bytes
```

Figure 7.4: LoRa Interface Real-time Monitor. The dashboard tracks the traffic flow between the Docker network (TESLA Packets) and the physical radio (LoRa Packets), visualising the packet fragmentation and reassembly events.

The `tesla-network` bridge driver is configured to emulate the local area network (LAN) constraints found on vessels. By abstracting the hardware dependency into a dedicated container, the architecture validates that the TESLA protocol remains agnostic to the underlying physical medium, interacting solely through standard maritime network interfaces (UDP Multicast/Unicast) as dictated by the IEC 61162-450 standard [18].

7.2.3 Simulation Modules: AIS Generator and VDES Bridge

To faithfully replicate the operational environment of a VDES station within the constraints of LoRa hardware, specific simulation modules were developed to handle data generation and hardware interfacing.

- AIS Generator Module (Network-Level Simulation):

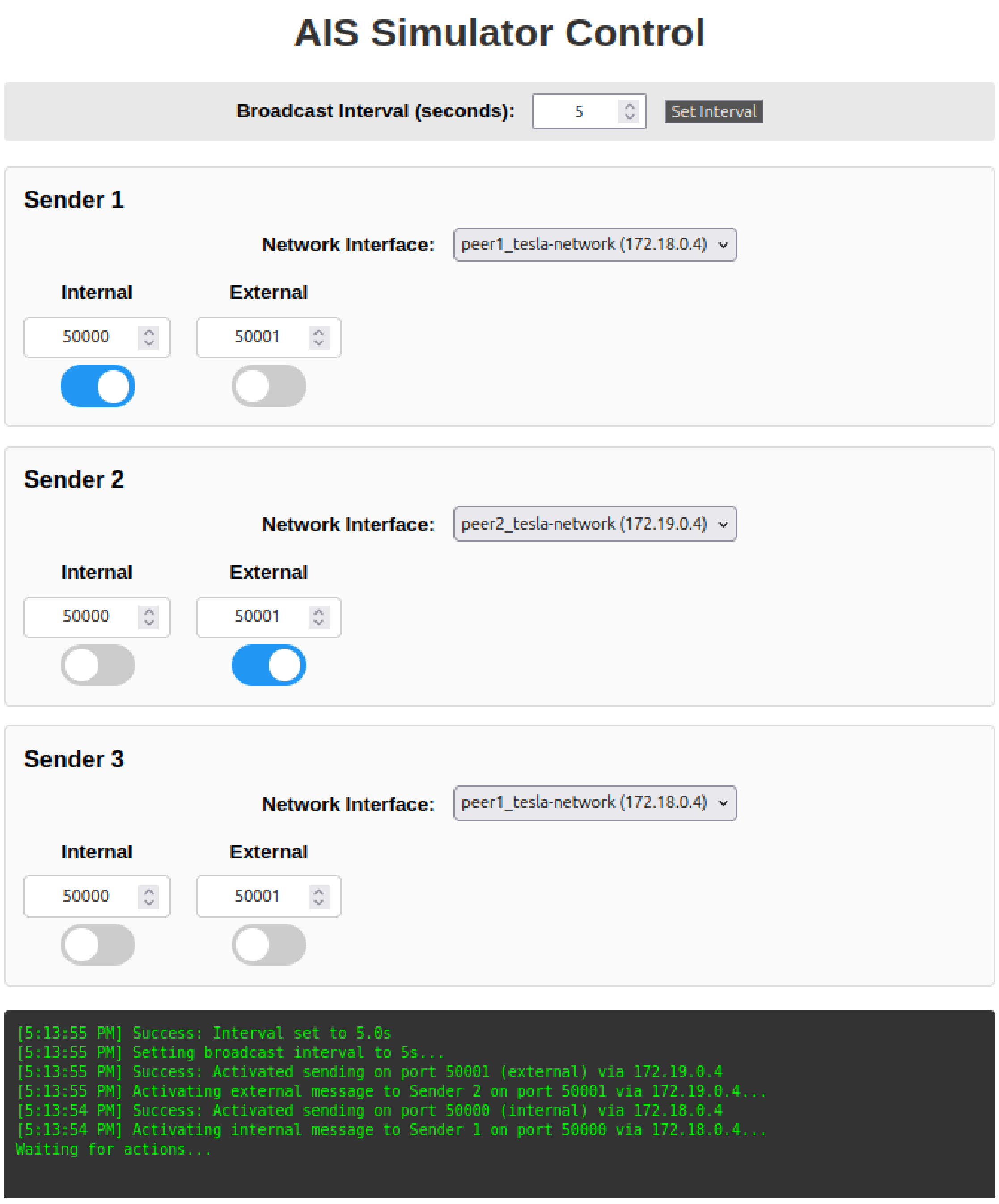
  In a real-world VDES deployment, standard AIS traffic and secure VDE-TER data operate on distinct physical frequency channels. However, since the LoRa development boards operate on a single shared channel, transmitting high-volume AIS traffic over the air would saturate the bandwidth available for the authentication protocol experiments [4].

  To overcome this physical limitation, the AIS Generator Module simulates the AIS data stream entirely via the Ethernet/UDP layer (IEC 61162-450). This module parses a static database of vessel movements (loaded via JSON configuration) and injects synthetic AIS NMEA messages into the Docker network. It features a granular configuration that allows the user to define target networks, destination ports, and transmission intervals.

  Crucially, it simulates two distinct logical flows via direct UDP injection into the TESLA module:

  1. *Internal Sensor Emulation:* Simulating data coming from the ship's own AIS.
  2. *External Traffic Emulation:* Simulating AIS messages that would theoretically be received by the VDES modem from other vessels.

  This approach ensures that the cryptographic protocol is tested with a realistic data load without consuming the limited airtime of the LoRa link.



AIS Simulator Control

Broadcast Interval (seconds): 5 Set Interval

Sender 1

Network Interface: peer1_tesla-network (172.18.0.4)

Internal	External
50000	50001
✓	

Sender 2

Network Interface: peer2_tesla-network (172.19.0.4)

Internal	External
50000	50001
	✓

Sender 3

Network Interface: peer1_tesla-network (172.18.0.4)

Internal	External
50000	50001

```
[5:13:55 PM] Success: Interval set to 5.0s
[5:13:55 PM] Setting broadcast interval to 5s...
[5:13:55 PM] Success: Activated sending on port 50001 (external) via 172.19.0.4
[5:13:55 PM] Activating external message to Sender 2 on port 50001 via 172.19.0.4...
[5:13:54 PM] Success: Activated sending on port 50000 (internal) via 172.18.0.4
[5:13:54 PM] Activating internal message to Sender 1 on port 50000 via 172.18.0.4...
Waiting for actions...
```

Figure 7.5: AIS Simulator Control Dashboard. The interface allows granular control over multiple simulated senders, enabling the configuration of broadcast intervals and specific network injection points (internal vs. external ports) to emulate complex traffic scenarios.

- VDES Bridge Module (Hardware Abstraction Layer):

  While AIS data are simulated over the network, the secure VDE-TER packets must traverse the physical radio link. This module acts as the gateway between the Dockerized IP network and the physical LoRa transceiver connected via

USB/Serial.

- *Transmission Path:* It listens for specific IEC 61162-450 UDP datagrams tagged for VDE transmission, strips the IP/UDP headers, and serialises the binary payload to the LoRa hardware.
- *Reception Path:* Listens to LoRa hardware for incoming radio packets (e.g., authentication handshakes), encapsulates them into valid IEC 61162-450 UDP datagrams, and multicasts them onto the internal Docker network.

This module renders the physical medium transparent, allowing the TESLA node to operate as if it were connected to a standard, multi-channel VDES modem.

7.2.4 The Secure Protocol Implementation Module

This module constitutes the core software artefact of the thesis. Developed in Python, it acts as a *Security Middleware*, implementing the execution logic of the adapted TESLA protocol defined in Chapter 6. Functionally, it operates as the *Auth Interceptor System* (see Figure 7.1), mediating the traffic between the ship's internal network and the external radio interface.

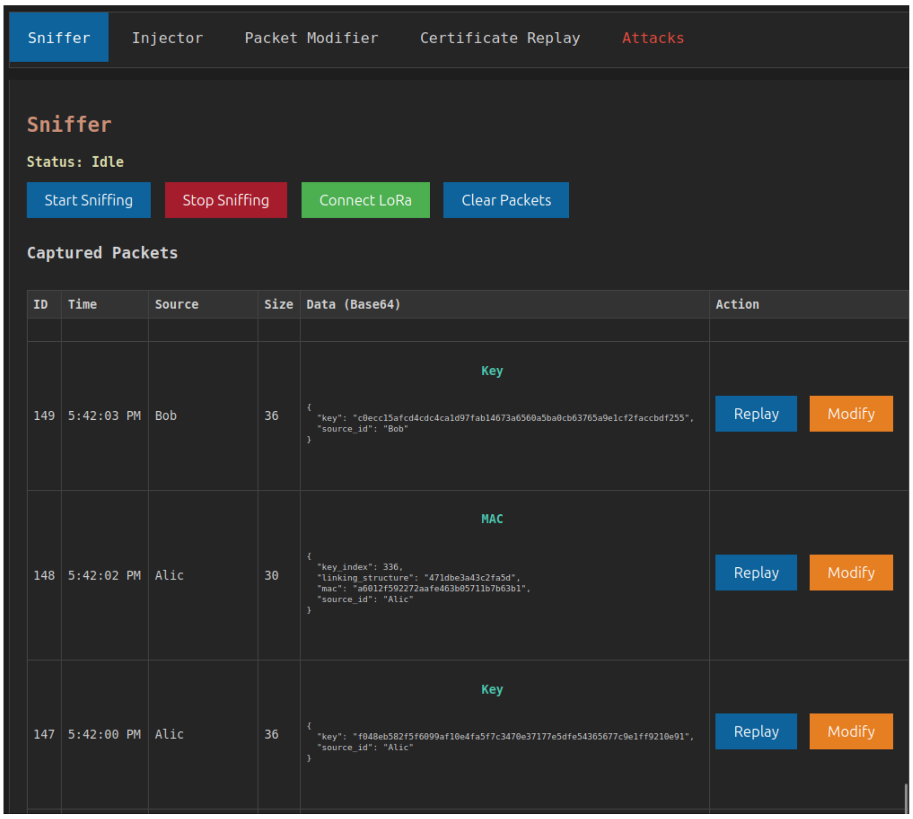
Its primary operational responsibilities include:

- **Cryptographic Execution:** It manages the generation and maintenance of the hash chains and executes the HMAC operations required to sign outgoing messages and verify incoming ones according to the protocol's security primitives.
- **Session and State Management:** It maintains the synchronisation state with peer stations, handling the strict timing requirements of the disclosure intervals and managing the transitions between protocol states.
- **Traffic Mediation:** It functions as a transparent proxy that intercepts generated AIS data to append authentication headers before transmission and buffers incoming radio packets to enforce the time-delayed verification mechanism before releasing them to the internal system.

7.2.5 Adversarial Testing Framework: The 'Evil_VDES' Module

To strictly validate the security properties of the proposed architecture beyond mere functional correctness, a dedicated offensive framework named `evil_vdes` was developed.

Deployed on an independent hardware node equipped with its own LoRa interface and network connection, this module acts as a "Red Team" appliance. It is



Sniffer

Status: Idle

Captured Packets

ID	Time	Source	Size	Data (Base64)	Action

149	5:42:03 PM	Bob	36	Key
				{
				"key": "c0ecc15afcd4cdc4ca1d97fab14673a6560a5ba0cb63765a9e1cf2faccbdf255",
				"source_id": "Bob"
				}
					Replay Modify

148	5:42:02 PM	Alic	30	MAC
				{
				"key_index": 336,
				"linking_structure": "471dbe3a43c2fa5d",
				"mac": "a6012f592272aafe463b05711b7b63b1",
				"source_id": "Alic"
				}
					Replay Modify

147	5:42:00 PM	Alic	36	Key
				{
				"key": "f048eb582f5f6099af10e4fa5f7c3470e37177e5dfe54365677c9e1ff9210e91",
				"source_id": "Alic"
				}
					Replay Modify

Figure 7.6: The Evil VDES Sniffer interface. The dashboard displays intercepted TESLA packets (key and MAC types) decoded in real-time, exposing actionable controls to replay or modify the payloads for attack simulation.

designed to simulate a sophisticated adversary capable of operating simultaneously on the physical layer (as an external attacker via radio) and the network layer (as a compromised onboard device via UDP). Controlled via a RESTful API and a web dashboard, `evil_vdes` orchestrates complex attack vectors to stress-test the resilience of the TESLA protocol:

- **Advanced Sniffing and Fragment Reassembly:**

  Unlike a passive logger, the module operates in promiscuous mode on both the IEC 61162-450 UDP bus and the LoRa serial interface. It implements logic to capture, decode, and crucially, *reassemble* fragmented LoRa packets. This allows the attacker to reconstruct high-level protocol messages (such as X.509 Certificate chains or TESLA key disclosures) to plan semantic attacks, rather than just raw byte capture.

- **Replay and Masquerading Attacks:**

  This vector tests the protocol's time-synchronisation enforcement. The module can record valid authentication packets (MACs/keys) and reinject them into the network after a configurable delay. It supports both *Raw Replay* (retransmitting the exact binary frame) and *Smart Injection* (repackaging intercepted payloads into new JSON structures), aiming to fool the receiver into accepting expired or duplicated commands.

- **Denial of Service (DoS) and Spectral Jamming:**

  The framework implements two distinct flooding strategies:

  - *Spectral Jamming:* Continuous transmission of high-entropy random bytes to physically saturate the LoRa channel, testing the system's ability to recover synchronisation after a blackout.
  - *Logic DoS:* Rapid injection of syntactically valid but cryptographically invalid packets. This targets the receiver's resources (CPU and RAM buffers), attempting to overflow the verification queue before legitimate keys can be processed.

- **Packet Forging and Tampering:**

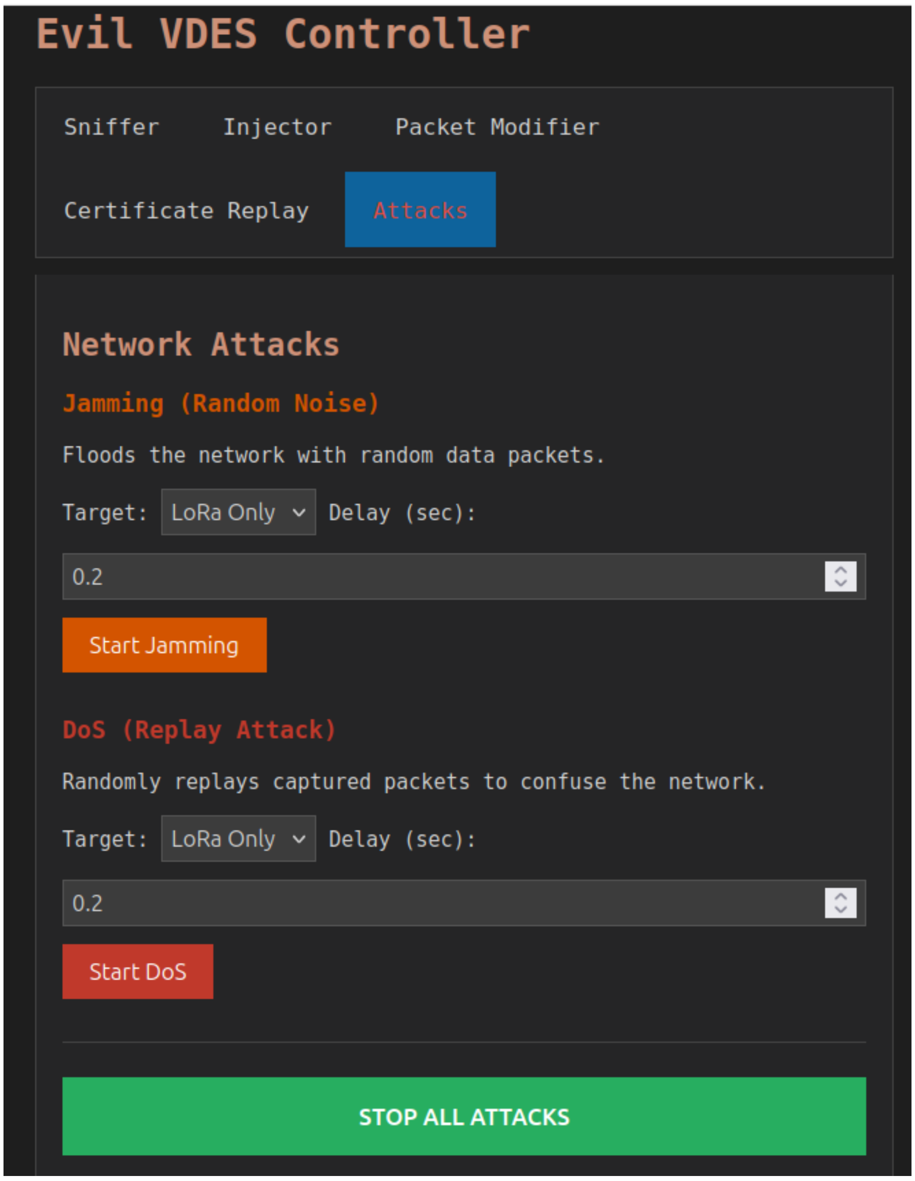
  Using the *Packet Injection* engine, the module can forge arbitrary TESLA packets with manipulated fields (e.g., spoofed Source IDs or modified MACs). This serves to verify integrity check failure rates and ensures that the protocol correctly discards unauthorised data traffic.

By exposing these capabilities via API, the `evil_vdes` module allows reproducible automated attack scenarios, the results of which are analysed in Chapter 9.

7.3 Interfacing with VDES Equipment According to IEC 61162-450 Standard

The implementation of an interoperable maritime communication architecture requires a standardised interface between the software system, which manages security logic (e.g., TESLA message generation), and hardware transmission equipment, such as VDES transceivers. The reference standard for this interconnection over IP networks is IEC 61162-450 "Lightweight Ethernet". This standard defines a modern high-speed protocol for data exchange between navigation devices, overcoming the limitations of traditional NMEA 0183 serial interfaces [18].

The prototype developed in this thesis adopts this standard to send data and authentication protocol messages to a VDES transceiver, which acts as a "gateway" for transmission over the VHF radio channel. The interface is based on two technical pillars defined by the standard: the network-level transport protocol and the specific structure of datagrams used for binary data transfer.



Network Attacks

Jamming (Random Noise)

Floods the network with random data packets.

Target: LoRa Only Delay (sec):

0.2

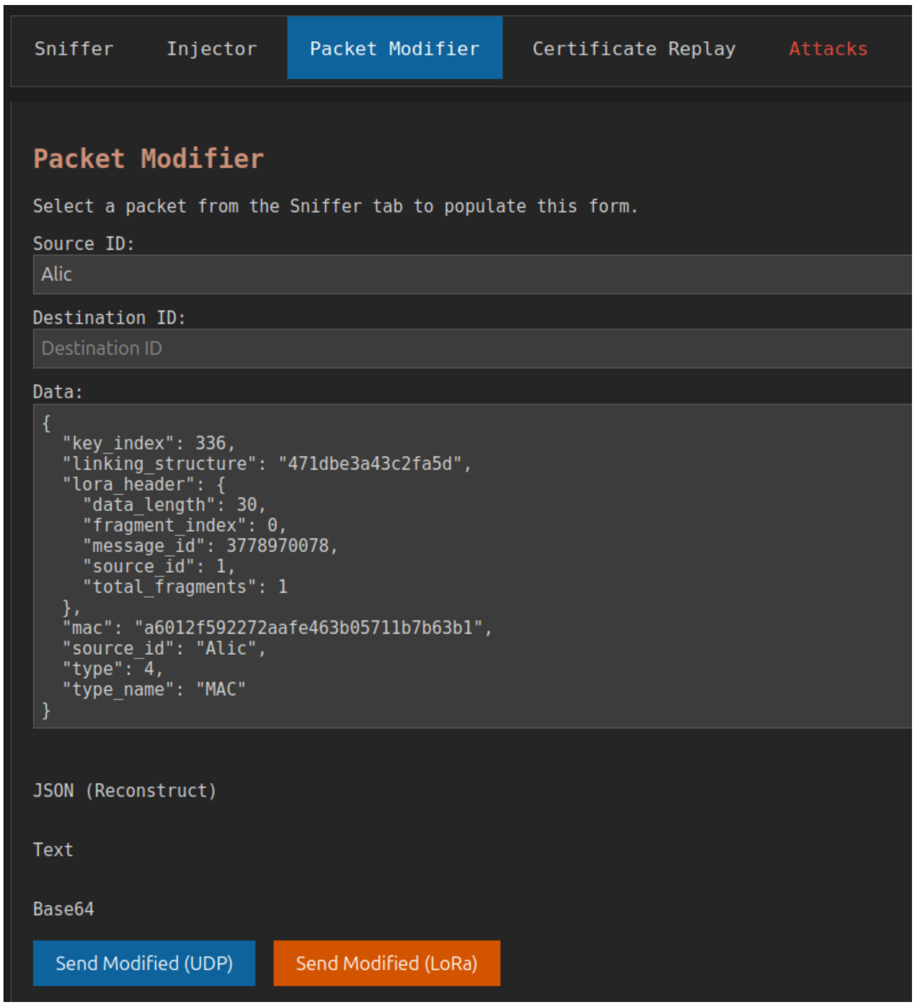
DoS (Replay Attack)

Randomly replays captured packets to confuse the network.

Target: LoRa Only Delay (sec):

0.2

Figure 7.7: Evil VDES Network Attack Interface. The panel provides controls to trigger specific availability attacks, allowing the operator to select the target layer (LoRa vs. Network) and configure the transmission delay for flooding intensity.



Packet Modifier

Select a packet from the Sniffer tab to populate this form.

Source ID: Alic

Destination ID: Destination ID

Data:

```
{
  "key_index": 336,
  "linking_structure": "471dbe3a43c2fa5d",
  "lora_header": {
    "data_length": 30,
    "fragment_index": 0,
    "message_id": 3778970078,
    "source_id": 1,
    "total_fragments": 1
  },
  "mac": "a6012f592272aafe463b05711b7b63b1",
  "source_id": "Alic",
  "type": 4,
  "type_name": "MAC"
}
```

Figure 7.8: Packet Modifier Interface. This tool allows the attacker to modify the internal structure of captured packets (e.g., altering the `mac` or `key_index` fields within the JSON payload) and re-inject them via LoRa or UDP to test the receiver's integrity validation logic.

7.3.1 Transport Protocol over IP Network: UDP Multicast

For the transport layer, the IEC 61162-450 standard specifies the use of UDP (User Datagram Protocol) in multicast mode. Unlike unicast communication, where packets are sent to a single recipient, multicast allows a single source to transmit data to a "group" of interested devices.

In this paradigm, the system does not send packets to a specific IP address of the transceiver but to a multicast IP address and designated port specified by the standard, on which all compatible devices (including the VDES transceiver) are listening. The standard reserves a specific address range for binary data transfer ("Simple Binary Image Transfer"), between 239.192.0.21 and 239.192.0.25, with corresponding ports in the range 60021-60025.

This architectural choice ensures efficiency and scalability, allowing multiple onboard systems (e.g., ECDIS, radar, monitoring systems) to simultaneously receive the same security data stream without the source system having to establish and manage multiple connections.

7.3.2 Datagram Structure for Binary Data Transfer (Binary Image Transfer)

The IEC 61162-450 standard explicitly defines the payload structure for each UDP datagram in addition to defining the transport protocol (as described in Section 7.3.1). Application data, in this case TESLA protocol messages, cannot be sent in raw format to guarantee interoperability. Rather, they have to be contained in a particular format known as "Binary Image Transfer" [18].

This structure ensures that any receiving device that is compliant with the standard can correctly interpret, reassemble, and validate the data received, regardless of its application nature. The general payload structure is sequential and composed of a maximum of four distinct sections:

```
[Token] + [Header] + [Binary Image Descriptor] + [Data Fragment]
```

The function and structure of each component are analysed in the following.

Token (Datagram Header)

Every UDP datagram must begin with a fixed 6-byte identifier (or "magic number"). Its purpose is to allow the receiver to immediately identify the type of data transfer in progress. For generic binary data transmission, the standard prescribes the use of the ASCII string `"RaUdP"` followed by a null character (`\x00`). This unique signature prevents misinterpretation of other types of NMEA datagrams that might coexist on the same network.

Header (Binary Block Header)

Immediately after the token follows a fixed-size binary header of 28 bytes. This section is crucial for the identification, routing, and reassembly of a data block should it need to be fragmented across multiple UDP packets. All multi-byte numeric fields within the header must be encoded in big-endian format (most significant byte first).

The structure, defined in Table 9 of the standard, is shown in Table 7.1.

Table 7.1: Header Field Structure

Field	**Type**	**Size**	**Description**
Version	WORD	2 bytes	Protocol version. Must be set to 1.
SrcID	STRING[6]	6 bytes	Unique identifier (ASCII) for the sending service (e.g., "TL0001").
DestID	STRING[6]	6 bytes	Destination station identifier. For multicast broadcast, use "XXXXXX".
Type	WORD	2 bytes	Message type. For application data, the value is 1 (DATA).
BlockID	DWORD	4 bytes	Unique identifier for the entire data block. All fragments of the same block must share this ID.
SequenceNum	DWORD	4 bytes	Packet sequence number within the block. For a single unfragmented packet, this value is 1.
MaxSequence	DWORD	4 bytes	Total number of packets composing the block. For a single packet, this value is 1.

Binary Image Descriptor

The Binary Image Descriptor, of variable length, provides metadata describing the underlying Data Fragment. It is a critical component, as it is present exclusively in the first packet of a data block (i.e., when the `SequenceNum` field in the header equals 1). Its structure is defined in Table 10 of the standard and is shown in Table 7.2.

Table 7.2: Binary Image Descriptor Structure

Field	**Type**	**Size**	**Description**
Length	DWORD	4 bytes	Total length in bytes of this descriptor section.
imageLength	DWORD	4 bytes	Total length in bytes of the entire binary data block (the complete, unfragmented payload).
Status	WORD	2 bytes	Indicates operation status. A value of 0 denotes normal operation.
Device / Channel	BYTE	1+1 bytes	Numeric identifiers of the source device and channel (e.g., 1 and 1).
TypeLength	BYTE	1 byte	Length of the following DataType string (including null terminator).
DataType	STRING[n]	n bytes	MIME string defining the data type. In the prototype, "application/octet-stream" is used.
Status text	STRING[n]	n bytes	Optional status text (can be an empty string terminated by `\x00`).

Data Fragment

This represents the terminal section of the datagram, designated to contain the actual application payload. In the context of this thesis, the data fragment corresponds to the logical payload of the TESLA protocol (e.g., MAC payload or key payload).

Crucially, the serialisation of these payloads is not performed using the JSON format. Instead, all TESLA messages are defined and managed as rigid, fixed-length binary structures through the custom TESLAProtocol module. The module utilises the Python struct library for efficient, low-level packing and unpacking of data types (come bytes, int, and float). This approach ensures maximum compactness and speed, which are vital for bandwidth-constrained maritime channels.

7.3.3 Mapping Protocol Messages to IEC 61162-450 Datagrams

To integrate the TESLA authentication protocol, defined in the logical layer, with the physical VDES transport channel, a rigorous serialisation and encapsulation strategy is essential. Each logical message generated by the system must be transformed into a fixed-length binary payload that is specifically formatted to ensure optimal transfer efficiency by the VDES Communication Module onto the designated radio channel, while remaining compliant with the IEC 61162-450 standard prior to transmission [18].

Unlike less efficient text-based formats, **JSON is not utilised**. The implementation relies instead on high-efficiency **binary serialisation** via the Python `struct` library. This approach ensures maximum compactness and processing speed, which are critical constraints for bandwidth-limited maritime channels. Furthermore, the use of specialised compression techniques (e.g., CXF for certificates) is employed to manage data volume within tight constraints.

The encapsulation process is expressed in the following steps:

1. **Binary Serialisation**: The logical data objects are converted into fixed-length byte sequences. This conversion adheres strictly to Network Byte Order and is formatted to a specific binary structure, the details of which are defined in the subsequent section (Section 7.4).
2. **Encapsulation (Packing)**: This binary sequence forms the Data Fragment (the application payload). Subsequently, it is inserted into the complete UDP datagram, preceded by the binary headers required by the IEC 61162-450 standard: [Token] + [Header] + [Binary Image Descriptor].

This structured approach ensures that the total size of the assembled UDP datagram respects the constraints necessary to prevent VDES-level fragmentation.

7.4 TESLA Protocol Packet Types

The implementation defines seven different message types to manage the authentication lifecycle, synchronisation, and key rotation within the system. For a detailed explanation of the role and function of the Announcement, Sync Request, and Sync Response packets within the initial trust establishment process, see Section 6.2.2.

7.4.1 Announcement (Type 1)

This is the first PKI-authenticated message transmitted by a station. It acts as a service discovery beacon, establishing an initial trust link between the sender and the receiver and carrying the essential identity and temporal data required for the Bootstrap phase [2].

- **Size Constraint**(Tab. 7.3):

  The payload has a fixed binary length to ensure predictability on the limited-bandwidth channel. The total size is fixed at 1137 bytes, with the following detailed structure:

Table 7.3: Announcement packet structure (size is in bytes)

Field	**Format**	**Size**	**Description**
Compressed Certificate	`869s`	869	Sender's MCP/MIR Certificate (compressed via CXF)
Digital Signature	`256s`	256	Signature of the payload
Source ID (source_id)	`4s`	4	Sender's identifier of 4 characters
Timestamp (t_r)	`d`	8	Transmission time (8-byte float)
		Total Payload Size	**1137 bytes**

7.4.2 Sync Request (Type 2)

This is a PKI-authenticated message sent by a sender to the verifier, which serves to initiate the authentication stream by establishing temporal alignment and committing the cryptographic root key.

- **Size Constraint**(Tab. 7.4):

  This is the largest packet type, with a fixed binary length of 1196 bytes, required by the large number of configuration parameters, the authentication data, and the commitment key.

Table 7.4: Sync Request packet structure (Size is in Bytes)

Field	**Format**	**Size**	**Description**
Source ID (source_id)	`4s`	4	Sender's identifier (Station B)
Peer Name (peer_name)	`4s`	4	Destination identifier (Station A)
Received Time (t_r)	`d`	8	Time of Received Announcement (proof of freshness)
Current Time (t_s)	`d`	8	Station B's current transmission time
Base Interval (base_interval)	`B`	1	Duration of one key validity interval (e.g., in seconds)
Buffer Packet (buffer_packet)	`B`	1	Required receiver buffer size (in packets)
Compressed Certificate	`869s`	869	Sender's MCP/MIR certificate (compressed via CXF)
Current Key Index (k_idx)	`H`	2	Index of the key currently in use by Station B
Hash Chain Length (h_len)	`H`	2	Total number of keys in the chain
Key Disclosure Delay (k_delay)	`B`	1	Time delay d before key revelation
Start Sending Time(start_sending_time)	`d`	8	Temporal start reference for the current key chain
Digital Signature	`256s`	256	Signature of the entire payload
Commit Key (commit_key)	`32s`	32	Root key of the current key chain ($\mathbf{K_0}$)
		Total Payload Size	**1196 bytes**

7.4.3 Sync Response (Type 3)

This is a PKI-authenticated message sent by the receiver to confirm the successful reception of the sync request and finalise the synchronisation. It acts as an acknowledgement (ACK) in the bootstrap sequence and provides the identifiers and temporal information required by the sender to complete synchronisation and compute the Round-Trip Time (RTT).

- **Size Constraint**(Tab. 7.5):

  The payload has a fixed binary length of 1141 bytes, reflecting the authentication data and the timestamp.

Table 7.5: Sync Response packet structure (Size is in Bytes)

Field	**Format**	**Size**	**Description**
Compressed Certificate	`869s`	869	Sender's MCP/MIR certificate (compressed via CXF)
Peer Name (peer_name)	`4s`	4	Destination identifier (Station B)
Digital Signature	`256s`	256	Signature of the payload
Source ID (source_id)	`4s`	4	Sender's identifier (Station A)
Current Time (t_s)	`d`	8	Station A's current transmission time
		Total Payload Size	**1141 bytes**

7.4.4 MAC Packet (Type 4)

This packet is the core operational message of TESLA, which contains the authentication code used to verify the AIS data packets transmitted in parallel. Unlike previous packet types, it is not PKI-signed; instead, it is authenticated symmetrically using the disclosed TESLA keys.

- **Size Constraint**(Tab. 7.6):

  The payload is intentionally compact, with a fixed binary length of 30 bytes, minimising the overhead on the constrained radio channel [8].

Table 7.6: MAC Packet structure (Size is in Bytes)

Field	**Format**	**Size**	**Description**
Key Index (key_index)	`H`	2	Index of the TESLA key used to generate the MAC.
Source ID (source_id)	`4s`	4	Sender's identifier (padded 4-character string).
Linking Structure	`8s`	8	The first 8 bytes of the data message hash are used to bind the MAC to the raw data packet.
MAC Bytes (mac_bytes)	`16s`	16	The computed Message Authentication Code.
		Total Payload Size	**30 bytes**

7.4.5 Key Packet (Type 5)

This packet carries the TESLA key disclosure message.

The disclosed key K_{i-d} allows the verifier to authenticate the MACs received during the previous interval, according to the temporal security delay d.

- **Size Constraint**(Tab. 7.7):

  The payload has a fixed binary length of 36 bytes.

Table 7.7: Key Packet Structure (Size is in Bytes)

Field	**Format**	**Size**	**Description**
Source ID (source_id)	`4s`	4	Sender's identifier (padded 4-character string).
Key Bytes (key_bytes)	`32s`	32	Disclosed symmetric key K_{i-d} used for MAC verification.
		Total Payload Size	**36 bytes**

7.4.6 Refill Packet (Type 6)

This packet is used for automatic TESLA re-keying; it contains the new keychain commitment root.

Integrity and authenticity are guaranteed by a MAC calculated with one of the final keys of the previous chain (transmitted separately as a MAC packet).

- **Size Constraint**(Tab. 7.8):

  The payload has a fixed binary length of 36 bytes.

Table 7.8: Refill Packet structure (Size is in Bytes)

Field	**Format**	**Size**	**Description**
Source ID (source_id)	`4s`	4	Sender's identifier (padded 4-character string).
New Commitment	`32s`	32	New keychain root commitment (K'_0).
		Total Payload Size	**36 bytes**

7.4.7 Encapsulation Example: MAC Packet inside an IEC 61162-450 Datagram

This example shows how IEC 61162-450 UDP datagrams are used to transport TESLA binary payloads. In particular, it demonstrates how the Data Fragment field of the datagram is filled in with the MAC Packet, the smallest and most common operational TESLA message.

The MAC Packet has a fixed size of 30 bytes and contains the key index, the source identifier, the linking structure, and the Message Authentication Code (MAC) [18].

Table 7.9: MAC Packet Encapsulation inside IEC 61162-450 Datagram

Section	**Field**	**Content / Value**	**Description**
Token	–	`b'RaUdP\x00'`	Fixed binary token used for UDP encapsulation.
Header (28 bytes)	Version	1	IEC 61162-450 protocol version.
	SrcID	`"TL0001"`	6-byte identifier of the sending TESLA module.
	DestID	`"XXXXXX"`	Broadcast or multicast destination for VDES networks.
	Type	1 (DATA)	Indicates application-level data.
	BlockID	12346	Unique identifier of the data block.
	SequenceNum	1	Sequence counter of this packet (first/only).
	MaxSequence	1	Total number of packets in this block (only 1).
Descriptor (variable length)	imageLength	30	Binary length of the embedded MAC packet.
	DataType	"application/octet-stream\x00"	Custom MIME type for TESLA Type 4 MAC packets.
	...	...	Additional mandatory descriptor fields (Status, Device, etc.).
Data Fragment	–	b'\x00\x01TL00\xAA\xBB\xCC\xDD...'	Serialised 30-byte MAC Packet (Type 4), encoded via `struct`.

7.5 Architectural Choices and Parameterisation

7.5.1 TESLA Parameter Selection

The correct configuration of the temporal and cryptographic parameters of the TESLA protocol is fundamental to ensuring both its security guarantees and its operational efficiency. In this work, three primary parameters were defined: the interval duration (Base Interval), the length of the hash chain (Hash Chain Length) and the disclosure delay (d). Their selection was guided by the need to balance computational overhead, verification timeliness, and resilience against forgery attacks [8].

Table 7.10: TESLA configuration parameters adopted in this work.

Parameter	**Value**
Base Interval	3 s
Hash Chain Length	28800 keys
Disclosure Delay	2 intervals

Base Interval:

The base interval (Base Interval) was set to 3 seconds. This value was strategically chosen to align with maritime operational constraints, particularly the validity period for AIS messages suggested by the IMO G.1192 standard (which recommends a validation time under 10 seconds) [2]. By setting the interval at 3 seconds and enforcing a minimum disclosure delay of $d = 2$ time slots (as further justified in Section 7.5.1), the protocol achieves a predictable verification window: The key for a message sent in interval i (K_i) is disclosed in interval $i + 2$. The total time elapsed until the key is available is $d \times$ Base Interval $= 2 \times 3\text{s} = 6$ seconds. The verification window therefore opens at a minimum of 3 seconds and closes at a maximum of 6 seconds (the end of the second interval, $2 \times 3\text{s}$). This timing ensures that the message is verified within the crucial 3-6 second range, allowing the protocol to effectively adhere to the suggested 10-second validity threshold for AIS messages while optimising the computational overhead. Although a shorter interval would reduce latency, the 3-second choice effectively balances the security delay requirement with the need for sustainable computational load and overhead for both sender and receiver.

Hash Chain Length

The length of the hash chain was set to 28,800 keys. This configuration was chosen to ensure uninterrupted operational coverage over a full 24-hour period. In combination with the 3-second base interval (as defined in the previous section), the total chain lifetime is given by:

$$T_{\text{chain}} = N \times T_{\text{interval}} = 28{,}800 \times 3 \text{ s} = 86{,}400 \text{ s} = 24 \text{ hours}.$$

This extended chain length enables prolonged operation between asymmetric PKI authentications and refill messages, reducing the frequency of high-cost cryptographic operations. Since each key is derived using SHA-256, the chain benefits from both computational efficiency and minimal storage requirements [14].

- **Memory Footprint:** Each SHA-256 key occupies 32 bytes. Therefore, the total memory consumption for a 28,800-element chain is:

  $$M_{\text{total}} = N \times 32 \text{ bytes} = 28{,}800 \times 32 = 921{,}600 \text{ bytes} \approx 0.88 \text{ MB}.$$

  This memory requirement is negligible for contemporary maritime embedded systems and is easily accommodated by standard hardware deployments.

- **Computation Time:** The total computation time required to pre-generate the hash chain is determined by the cumulative cost of each hashing operation. The overall generation time is expressed by the following formula:

  $$T_{\text{tot}} = N \times t_{\text{hash}},$$

  where:

  - T_{tot} denotes the total computation time (in seconds),
  - $N = 28{,}800$ is the chain length,
  - t_{hash} is the average computation time for SHA-256 per hash (in seconds/hash).

  Modern mid-range processors are capable of computing SHA-256 at rates of several thousand hashes per second. Using a measured reference throughput of approximately 5.36 kH/s (based on an *Intel i5 10th-generation processor*), the per-hash computation time is:

  $$t_{\text{hash}} = \frac{1}{5{,}360} \text{ s} \approx 1.87 \times 10^{-4} \text{ s/hash}.$$

  Applying this realistic value to the complete 28,800-element hash chain yields:

  $$T_{\text{tot}} = \frac{28{,}800}{5{,}360} \text{ s} \approx 5.37 \text{ s}.$$

Thus, the total time required to generate the entire 28,800-key chain is approximately 5.4 s, representing a modest and fully acceptable computational cost during system bootstrap. This configuration therefore achieves an optimal balance between long-duration serviceability, affordable computation time, and minimal memory requirements, thus contributing to the overall robustness and operational practicality of the TESLA deployment.

Disclosure Delay

The disclosure delay (d) was set to $d = 2$. This extended delay represents a deliberate architectural choice aimed at mitigating the temporal uncertainty introduced by the Round-Trip Time (RTT) in the maritime VDES communication environment. Due to the variable distances between communicating stations, the RTT can fluctuate significantly, thus creating a *gray zone* (ΔT_{grey}), a period of uncertainty in message reception, near the boundary of each time slot (Base Interval).

In this context, a DATA or MAC packet transmitted precisely at the end of the slot i may, depending on the variation of the RTT, be received at the end of slot i or at the beginning of slot $i + 1$. This variability introduces ambiguity in assigning the received message to its correct originating interval.

To address this challenge, two complementary design principles are applied:

- **RTT Independence:** The system is designed to remain agnostic to RTT-induced uncertainty. Each message is assigned solely to the time slot in which it is received, without compensating for propagation delays.
- **Safety Margin for Propagation:** Employing a disclosure delay of $d = 2$ ensures that, by the time key K_i is revealed at the beginning of interval $i + 2$, sufficient time has elapsed for all packets transmitted during slot i to reach their destination. This safety margin of an entire slot ($i + 1$) enables correct verification even in the presence of significant network delays or jitter, ensuring that legitimate packets are not prematurely discarded.

More precisely, when K_i, the key used to generate the MACs during interval i, is disclosed at the start of interval $i + 2$, the receiver performs two parallel checks:

1. Attempts to authenticate all buffered MACs associated with the interval i using K_i.
2. It simultaneously authenticates MACs stored in the interval $i+1$ with the same key K_i.

This mechanism ensures robustness against grey-zone effects. Even if a message produced with K_i is assigned erroneously to the buffer of slot $i + 1$ due to the uncertainty of reception, it is still verifiable using the correct key. Consequently, the system maintains its integrity and authentication correctness under dynamic and realistic operating conditions.

7.5.2 MAC and Linking Structure Truncation

The decision to truncate specific fields within the MAC packet is directly driven by the strict payload limitations imposed on VDES channels. In particular, the design must ensure that all non-certified packets can be transmitted using Message Types 92 and 93 on Link 11, the most bandwidth-constrained VDES link. According to the maximum-payload table presented in the *IALA Guideline G1192 – VDES Authentication Techniques* (Table 2, p. 16) (table 7.11), Link 11 supports significantly smaller payload sizes compared to terrestrial wideband links [2]. This constraint requires a highly compact encoding of authentication data to prevent fragmentation and maintain channel efficiency.

Table 7.11: Maximum message payload size (bit) by message type for VDE-TER

VDE Message Type	**Message Name**	**VDES Link ID**		
		11	**17**	**19**
74	Start fragment	280	1720	5464
75	Continuation fragment	280	1720	5464
76	End fragment	280	1720	5464
92	Short data message (ACK)	296	1736	5480
93	Short data message (no ACK)	304	1744	5488

Linking Structure Truncation

The Linking Structure field is truncated to 8 bytes to comply with the payload restrictions described above. A full cryptographic hash such as SHA-256 (32 bytes) would exceed the allowed size budget for Message Types 92/93 in Link 11, especially when combined with the remaining fields of the MAC Packet. By reducing the field to 8 bytes (64 bits), the design guarantees that complete, non-certified messages fit entirely within a single transmission on the narrowband Link 11 channel, as required by the constraints reported in the G1192 payload table [2].

Despite the truncation, a 64-bit hash still provides: $2^{64} \approx 1.8 \times 10^{19}$ possible values, which yields an acceptably low accidental-collision probability for operational maritime communication scenarios. The truncation therefore represents a calculated trade-off: it substantially decreases overhead while preserving the core function of the hash, binding each MAC to its corresponding message.

MAC Truncation

The Message Authentication Code is computed using the full 32-byte output generated by HMAC-SHA-256, but only the first 16 bytes (128 bits) are transmitted. This decision ensures that the MAC Packet does not exceed the payload limits specified for Message Types 92/93 on Link 11 (Table 7.11). As highlighted in the G1192 document, oversized authentication data may require fragmentation, which is undesirable for bandwidth-constrained channels and reduces system reliability in high-load maritime environments.

Upon reception, the verifier recomputes the complete 32-byte MAC and compares its first 16 bytes with the received truncated MAC. Only if these 16 bytes match is the packet accepted.

Cryptographic Security Basis:

The security remains anchored in the full 32-byte internal HMAC computation and in the secrecy of the underlying TESLA key, disclosed only after the temporal delay. The transmitted truncation merely limits the portion exposed over the channel.

Resistance to Brute-Force Forgery:

A 128-bit transmitted MAC preserves a search space of:

$$P(\text{success}) = \frac{1}{2^{128}},$$

which is computationally infeasible for an adversary without the disclosed TESLA key.

Efficiency and Compliance:

The 50% reduction from the full 32-byte MAC to a 16-byte transmitted MAC is essential to stay within the strict payload limits of Link 11, as evidenced in Table 7.11. This ensures that MAC packets remain unfragmented, compact, and compatible with legacy VHF constraints while maintaining adequate security. The 32-byte full-length MAC is still used internally, ensuring robust cryptographic validation despite truncated transmission.

Chapter 8

Performance Evaluation: Computational Cost and Latency

This chapter gives a *analytical performance evaluation* of the proposed authentication architecture. The analysis is not based on direct measurements on a live VDES system, but is instead *parameterised* using a combination of fragmentation and bandwidth feasibility results derived in Chapter 7, timing constraints defined in VDES technical standards, and documented cryptographic processing benchmarks obtained from a reference hardware platform.

Specifically, the fragmentation requirements for PKI-protected messages ($N_{frag} = 4$) and the atomic transmission property of TESLA messages ($N_{frag} = 1$) are obtained straight from the feasibility study conducted in Chapter 7. Cryptographic verification times are estimated assuming a reference Intel Core i5 (10th generation) processor, consistent with published benchmarks and reflective of commodity hardware deployed in shore-side Vessel Traffic Service (VTS) infrastructure. No real-time VDES radio measurements were taken, as access to the operating equipment was not available within the project duration.

Following the bandwidth feasibility analysis, which showed fragmentation as a crucial limiting factor for PKI-based authentication, this chapter tackles the remaining two essential characteristics of system performance: **computational overhead** and **authentication latency**.

The purpose is to examine the operating behaviour of both shore-side receivers and onboard units in high-load situations, applying a cautious, doubt-driven methodology. First, the theoretical scalability of the suggested design is questioned; second, its viability is tested under actual, literature-backed operating assumptions.

8.1 Computational Overhead Analysis (O_{cpu})

In a maritime broadcast environment, computational asymmetry constitutes a significant risk factor. Although warships generate authorised messages at relatively low rates, a VTS shore station must validate signals emanating from hundreds of transmitters continuously. Consequently, even minor per-message verification fees can become a severe processing bottleneck.

8.1.1 Analytical Model

Let $C(M)$ be the computing cost associated with confirming a single authenticated message M.

Pure PKI Model (Asymmetric Cryptography) In the PKI-based system, the major cost originates from the verification of an ECDSA digital signature, which involves elliptic curve point multiplication.

$$C_{PKI} = T_{hash}(M) + T_{verify\_sig}(S, K_{pub})$$

Where T_{verify_sig} is mathematically complex ($O(n^3)$ or similar depending on implementation) [22].

TESLA Model (Symmetric Cryptography) The verification cost involves only hash chain traversal and HMAC re-computation.

$$C_{TESLA} = T_{hash}(M) + T_{HMAC}(M, K_i) + T_{hash}(K_i)$$

All operations are bitwise or modular additions, executed in linear time $O(n)$ [9].

8.1.2 Deduction: Computational Scalability in High-Density Scenarios

Although a cryptographic verification latency of a few milliseconds may appear small in isolation, system feasibility must be weighed against the collective workload of a shore-side receiver.

Assuming a high-density marine environment, such as a major commercial port, a VTS station may be required to process authenticated communications from around $N = 500$ vessels. Under a worst-case assumption in which each vessel transmits one authenticated message every second, the computing load scales linearly with N.

Based on established cryptographic benchmarks for an Intel Core i5 (10th generation), we assume the following representative verification times:

- **PKI Computational Load:**

  Given a verification time of $\tau_{PKI} \approx 4.1\,\text{ms}$ per signature, the aggregate processing time required per second is:

$$\text{Load}_{PKI} = 500 \times 4.1\,\text{ms} = 2.05\,\text{s}$$

This result implies that the required processing time exceeds the available physical time, resulting in CPU saturation (load $> 200\%$). Consequently, the receiver is forced to drop valid packets solely due to processing latency, effectively creating a vulnerability to computational Denial of Service (DoS).

- **TESLA Computational Load:**

  In contrast, with a verification time of $\tau_{TESLA} \approx 0.015\,\text{ms}$, the aggregate load becomes:

  $$\text{Load}_{TESLA} = 500 \times 0.015\,\text{ms} = 7.5\,\text{ms} = 0.0075\,\text{s}$$

  In this configuration, CPU utilisation remains below 1%. This shows that the lightweight nature of TESLA renders the system practically immune to computational flooding, ensuring scalability even in saturated network conditions [9].

8.2 Authentication Latency Analysis (T_{auth})

This section studies the latency between the physical transmission of a message and its successful cryptographic authentication. The analysis clearly integrates the fragmentation requirements specified in Chapter 7 and the temporal limits defined in VDES technical documentation.

8.2.1 Latency Composition

PKI Latency (Transmission-Bound)

For PKI, the latency is not structural but physical. Since the previous chapter established that a signed message requires $N_{frag} = 4$ fragments on Link 11, latency is the time required to flush these fragments into the radio channel plus the CPU time.

$$T_{auth\_PKI} = (N_{frag} \times T_{slot}) + T_{CPU\_Verify}.$$

Assuming a VDES time-slot duration of $T_{slot} \approx 26.6\,\text{ms}$, as specified in the technical documentation, and using the previously stated CPU verification time:

$$T_{auth\_PKI} \approx (4 \times 26.6\,\text{ms}) + 4.1\,\text{ms} \approx 110\,\text{ms}.$$

TESLA Latency (Disclosure-Bound)

TESLA authentication eliminates fragmentation by transmitting authentication material within a single atomic packet ($N_{frag} = 1$). However, verification is delayed by the protocol-enforced disclosure delay d, represented in time intervals of period T_{int} [8].

$$(d-1) \cdot T_{int} < T_{auth\_TESLA} \leq d \cdot T_{int}$$

Consequently, the latency is defined as a range rather than a scalar value, strictly dependent on the chosen configuration parameters. Adopting the robust parameters defined on the testbed ($T_{int} = 3s$, $d = 2$):

$$T_{auth\_TESLA} \in [\mathbf{3}\ \mathrm{s}, \mathbf{6}\ \mathrm{s}]$$

Specifically, a message transmitted at the end of the interval experiences the minimum latency ($3s$), while a message transmitted at the beginning must wait for the full duration of the current slot plus the disclosure delay ($6s$).

8.2.2 Critical Deduction: The Timeliness Trade-off

A potential concern regarding the use of TESLA in maritime collision-avoidance scenarios is related to its inherent verification delay. Since TESLA requires a disclosure interval, set in our design to 6 s, one may argue that such latency is incompatible with time-critical navigation tasks, especially when compared to the nominal 0.11 s verification time of a PKI-based scheme. At first glance, this discrepancy appears to undermine the operational safety of TESLA. However, this comparison is misleading unless contextualised within the characteristics of the VDES communication channel defined by technical standards [4].

The Theoretical Nature of PKI's "Instant" Verification

The commonly cited 0.11 s latency for PKI corresponds to the transmission time of a fully assembled message. Crucially, it assumes that all four fragments that make up the PKI-protected payload are received without error. To model a realistic environment, we derive channel reliability from the *Draft IEC Technical Specification for VDES* (March 2017) [5]. The specification defines acceptable Packet Error Rate (PER) limits ranging from 1% for high input levels (Table 9) to 20% at the receiver sensitivity limit (Table 8).

By adopting a nominal average PER of 10% (the approximate mean between the specified extremes), the probability of receiving all four fragments required for PKI verification is:

$$P_{\text{success}} = (1 - \text{PER})^4 = 0.9^4 \approx 0.656.$$

Thus, only about two-thirds of the transmitted messages can be verified within the nominal 0.11 s. If even a single fragment is lost, the message becomes unrecoverable due to the absence of ARQ mechanisms in the broadcast channel. The receiver must therefore wait for the next complete transmission cycle, typically on the order of 10 s for standard AIS reporting intervals [3]. In this context, PKI provides fast verification only for the subset of messages that survive fragmentation, while effectively yielding unbounded latency (i.e., data loss) for the remainder.

The Deterministic Reliability of TESLA

TESLA introduces a verification delay of between 3 and 6 seconds, but crucially transmits authentication material within a single atomic packet. Under the same derived average conditions (PER = 10%), the probability of successful reception is:

$$P_{\text{success}} = 1 - \text{PER} = 0.9.$$

Although slower than in the idealised PKI case, TESLA's behaviour is deterministic and substantially more robust to transmission impairments. Moreover, the imposed delay must be interpreted within the operational tempo of the maritime situational awareness systems. According to ITU-R M.1371, AIS position reports are broadcast every 2 to 10 s depending on vessel speed and manoeuvring state. Thus, a 6 s authentication delay therefore remains fully contained within the natural refresh cycle of the maritime tactical picture [3].

Conclusion

The comparison between TESLA and PKI highlights a fundamental design trade-off: the "fragile speed" of PKI versus the "robust slowness" of TESLA. PKI offers minimal verification latency but suffers significantly from fragmentation, leading to frequent irrecoverable losses. TESLA, by contrast, provides delayed but predictable and highly reliable authentication. In the context of maritime navigation, where periodic updates inherently introduce temporal smoothing, a verified but slightly delayed message is operationally preferable to an unverified and potentially missing one. Consequently, TESLA's verification delay is not only acceptable but also aligned with the practical requirements of the VDES-based maritime safety ecosystem.

Chapter 9

Security Analysis and Threat Modelling

The proposed architecture operates in a hostile environment where the attacker is assumed to have full control over the wireless channel (Dolev-Yao model) with the capability to intercept, modify and inject messages [12]. The following analysis looks at how strong the protocol is against certain types of attacks. Threats are categorised as either *mitigated* or *residual* depending on the existence of explicit and protocol-level countermeasures.

If the suggested architecture has a real cryptographic or architectural mechanism that stops the threat from being successfully exploited under the hypothetical attacker model, it is said to be "mitigated." On the other hand, "residual threats" are attack vectors that cannot be completely neutralised by protocol design alone and whose effects are mitigated but not completely gone, usually because of differences in physical layers or resources.

9.1 Mitigated Attacks

9.1.1 Cryptographic Collision on Linking Structure

A structural vulnerability identified in early design stages concerns the collision resistance of the truncated hash used to link Data Packets (AIS) with MAC Packets (VDE-TER).

While a full SHA-256 output provides 128-bit collision resistance, the 64-bit truncation offers a security margin of approximately $O(2^{32})$ operations against a Birthday Attack. In the specific context of maritime communications, where the validity window of a message is constrained to a few seconds (time slot), generating a meaningful collision in real-time is computationally impractical. Consequently, the probability of an attacker successfully injecting a malicious AIS payload that matches the linking structure of a legitimate authenticated packet is considered negligible [11].

9.1.2 Time-Shifting and Replay Attacks

The protocol is inherently resilient to replay attacks due to the strict temporal coupling between the key index (K_i) and the absolute time derived from the station' clocks. An adversary attempting to re-transmit a valid, previously recorded pair of data and MAC packets would fail, as the receiver would attempt verification using a current key index (K_{i+n}) incompatible with the stale MAC [8].

Furthermore, time-shifting attacks, where an adversary attempts to desynchronise the receiver during the bootstrap phase, are mitigated by including signed timestamps (t_{rA}, t_{sB}) within the PKI-authenticated handshake. The *Safe Packet Test* effectively discards any message that carries a key index that the receiver perceives as already disclosed, neutralising attempts to exploit network latency or manipulated timestamps [9].

9.1.3 Wormhole Attacks (Tunnelling)

Although wormhole attacks are typically a critical threat in broadcast protocols, the specific temporal parameters adopted in the proposed architecture ($d = 2$) impose severe constraints on tunnelling feasibility, effectively mitigating the threat. The receiver implements a strict *Acceptance Window*: a packet generated in time slot i is accepted for buffering only if received within slots i or $i+1$, in order to accommodate legitimate RTT variance. The disclosure key K_i is subsequently broadcast at the beginning of the slot $i + 2$.

Consequently, any tunnel attempt that introduces a transmission latency that exceeds the duration of the slot $i+1$ results in the packet being immediately discarded as obsolete ($t_{rx} > t_{validity}$), well before the key verification phase.

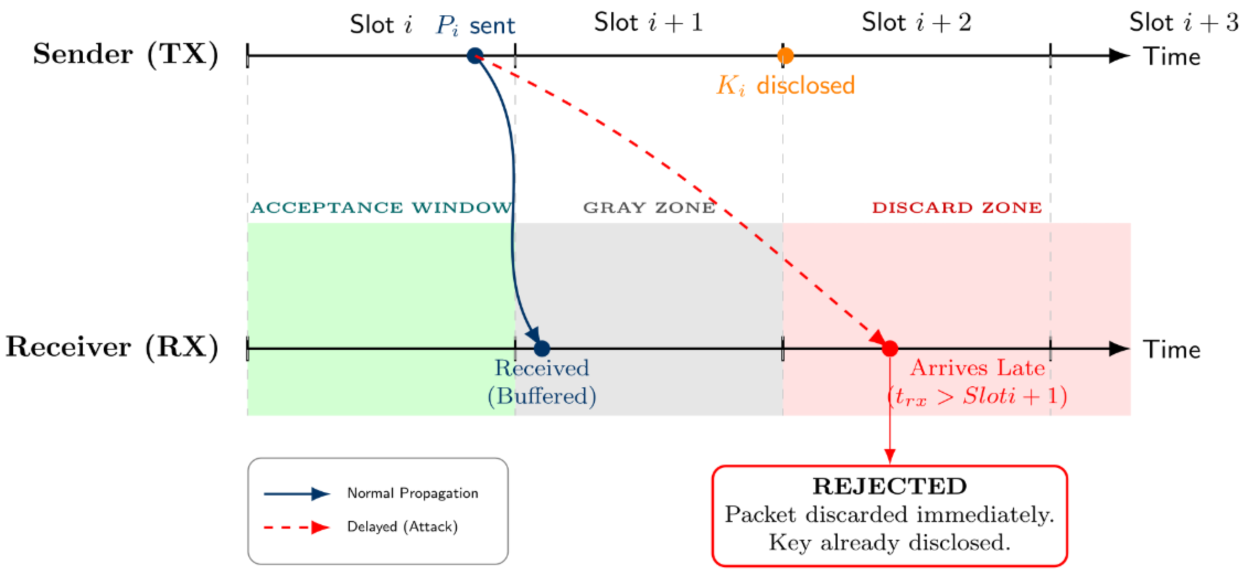


Figure 9.1: Mitigation of Wormhole Attacks via Time-Slot Acceptance Window. Legitimate packets (blue) arrive within slots i or $i+1$ (gray zone)and are buffered. Tunnelled packets (red), delayed by the attack infrastructure, arrive after the cutoff (start of slot $i+2$) and are discarded as obsolete before key disclosure (K_i) occurs in slot $i+2$.

9.1.4 Quantum Threats (Operational Phase)

The high-frequency operational phase relies exclusively on symmetric primitives (SHA-256 for the one-way key chain and HMAC-SHA256 for message authentication). These primitives are intrinsically resistant to Shor's algorithm. While Grover's algorithm theoretically halves the effective key space, the employed key length (256 bits) maintains a security level of 128 bits, which is sufficient to ensure post-quantum resistance for the data stream without requiring architectural modifications [17, 16].

9.2 Residual threats and Open Challenges

9.2.1 Asymmetric Resource Exhaustion and Channel Saturation (Bootstrap Phase)

The bootstrap phase presents a structural asymmetry that exposes the shore station to Denial of Service (DoS) attacks. The *Sync Response* packet requires the generation of a digital signature by the authority. An attacker can flood the station with validly formatted *Sync Request* messages, forcing the station to expend significant CPU cycles for signature generation, while the attacker incurs a negligible computational cost [13].

More critically, such a flood inevitably saturates the bandwidth of the communication channel. Since VDE-TER operates on a shared medium with limited capacity, the high volume of malicious traffic results in spectrum congestion and packet collisions, making the station physically unable to receive legitimate messages [4]. Although computational load can be mitigated through rate-limiting strategies, the denial of service at the physical layer represents an inherent limitation of the narrowband radio channel that cannot be resolved through software or architectural countermeasures.

9.2.2 Buffer Flooding, Channel Saturation, and Key Withholding

The "out-of-band" nature of the architecture necessitates a complex dual-buffering mechanism at the receiver, exposing the system to both memory exhaustion and bandwidth saturation attacks. An adversary can flood the radio channels with spurious packets to exploit this structure.

Beyond the inevitable saturation of the physical narrowband channel, which causes collisions and prevents the reception of legitimate signals, the memory management logic presents a specific vulnerability due to the decoupling of data and authentication. Although MAC packets are explicitly linked to a single source ID and are buffered accordingly, standard AIS data packets lack authenticated identity upon reception. Consequently, the receiver is forced to buffer each data packet and speculatively associate it with the current key indices of *every* active synchronised peer. This multiplicative indexing significantly amplifies the memory footprint required for each received message [9].

A persistent flood can deplete the receiver's RAM before these expiration conditions are satisfied, despite the system's garbage collection policy, which purges data packets only when their associated key indices become outdated for all possible peers. Similarly, a compromised sender could perform a *Key Withholding* attack, broadcasting authenticated data streams but deliberately withholding the Key Packets, forcing receivers to keep these speculative data entries in the buffer until the key index attached to the MAC becomes obsolete for the peer.

9.2.3 Selective Jamming and Security Downgrade

The architecture relies on the simultaneous reception of two distinct radio channels. An adversary employing a Software Defined Radio (SDR) can perform *Selective Jamming*, targeting exclusively the VDE-TER frequency used for authentication, while leaving the legacy AIS channels undisturbed [2].

In this scenario, the receiver successfully acquires the AIS data but fails to receive the corresponding authentication metadata. Although the system correctly flags these messages as "Unverified," this effectively forces a downgrade to the security level of the legacy system. If onboard operators or downstream systems are configured to accept unverified data to ensure operational continuity, the security layer is effectively bypassed.

9.3 Experimental Security Validation

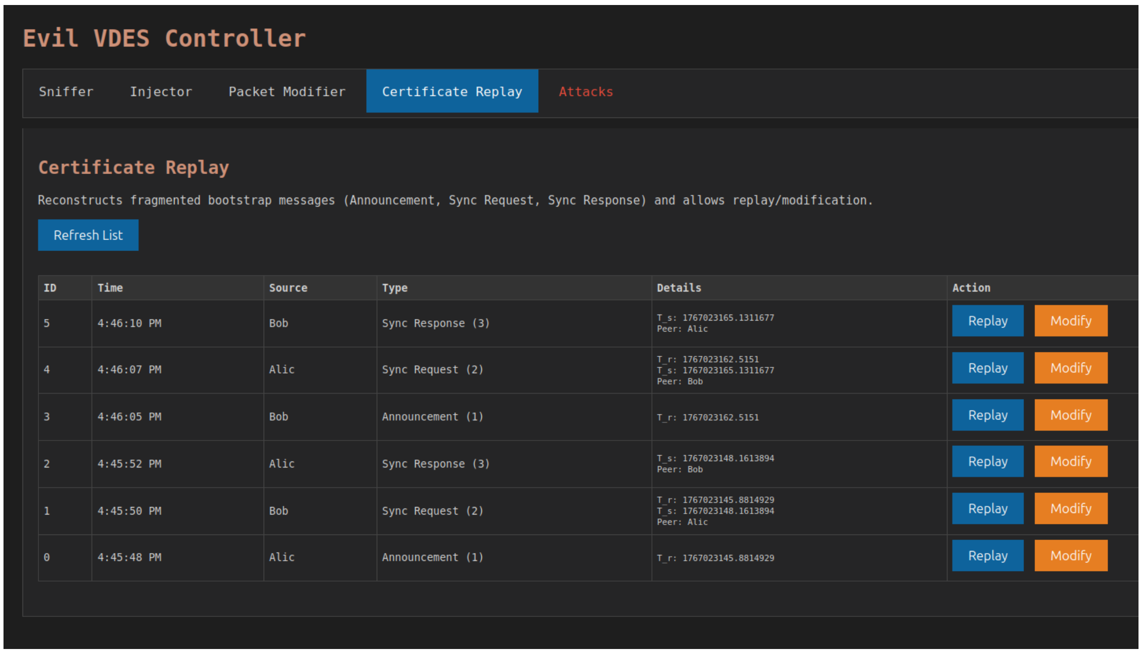
To validate the theoretical assertions presented in the previous sections, a specific adversarial testing campaign was conducted using the LoRa-based testbed and the `evil_vdes` framework described in Chapter 7. The experimental setup consisted of three active nodes operating in a broadcast environment:

- **Alice (Station A):** A legitimate sender broadcasting authenticated navigation data.
- **Bob (Station B):** A legitimate receiver implementing the TESLA verification logic.
- **Eve (The Attacker):** A malicious node running the `evil_vdes` module, capable of sniffing, jamming, and injecting arbitrary packets.

The following subsections detail the results of the three primary attack vectors executed against the protocol.

9.3.1 Test Case 1: Bootstrap Integrity and Replay Resistance

This test aimed to validate the protocol's resistance against manipulation and replay attacks during the critical Trust Establishment phase. The adversarial node, configured with the *Evil VDES Controller* (see Section 7.5), was used to intercept the full handshake sequence between legitimate stations (Alice and Bob) and attempt unauthorized re-injection.



Certificate Replay

Reconstructs fragmented bootstrap messages (Announcement, Sync Request, Sync Response) and allows replay/modification.

ID	Time	Source	Type	Details	Action
5	4:46:10 PM	Bob	Sync Response (3)	T_s: 1767023165.1311677 Peer: Alic	Replay Modify
4	4:46:07 PM	Alic	Sync Request (2)	T_r: 1767023162.5151 T_s: 1767023165.1311677 Peer: Bob	Replay Modify
3	4:46:05 PM	Bob	Announcement (1)	T_r: 1767023162.5151	Replay Modify
2	4:45:52 PM	Alic	Sync Response (3)	T_s: 1767023148.1613894 Peer: Bob	Replay Modify
1	4:45:50 PM	Bob	Sync Request (2)	T_r: 1767023145.8814929 T_s: 1767023148.1613894 Peer: Alic	Replay Modify
0	4:45:48 PM	Alic	Announcement (1)	T_r: 1767023145.8814929	Replay Modify

Figure 9.2: The 'Evil VDES Controller' interface capturing the bootstrap sequence. The tool exposes 'Replay' and 'Modify' vectors for Announcement, Sync Request, and Sync Response packets.

The analysis focused on two specific attack vectors:

Vector A: Message Modification and Injection

The attacker utilized the "Modify" function to alter the payload of captured packets (e.g., manipulating the Source ID in a *Sync Request*) before re-injecting them.

Result: The attack was structurally futile. Since every bootstrap message is digitally signed using the sender's private key and carries the associated X.509 certificate, any modification to the binary payload, however minor, invalidates the digital signature. The receiver's cryptographic engine correctly detected the integrity violation and discarded the modified packets immediately, logging a `Signature Verification Failed` error [13].

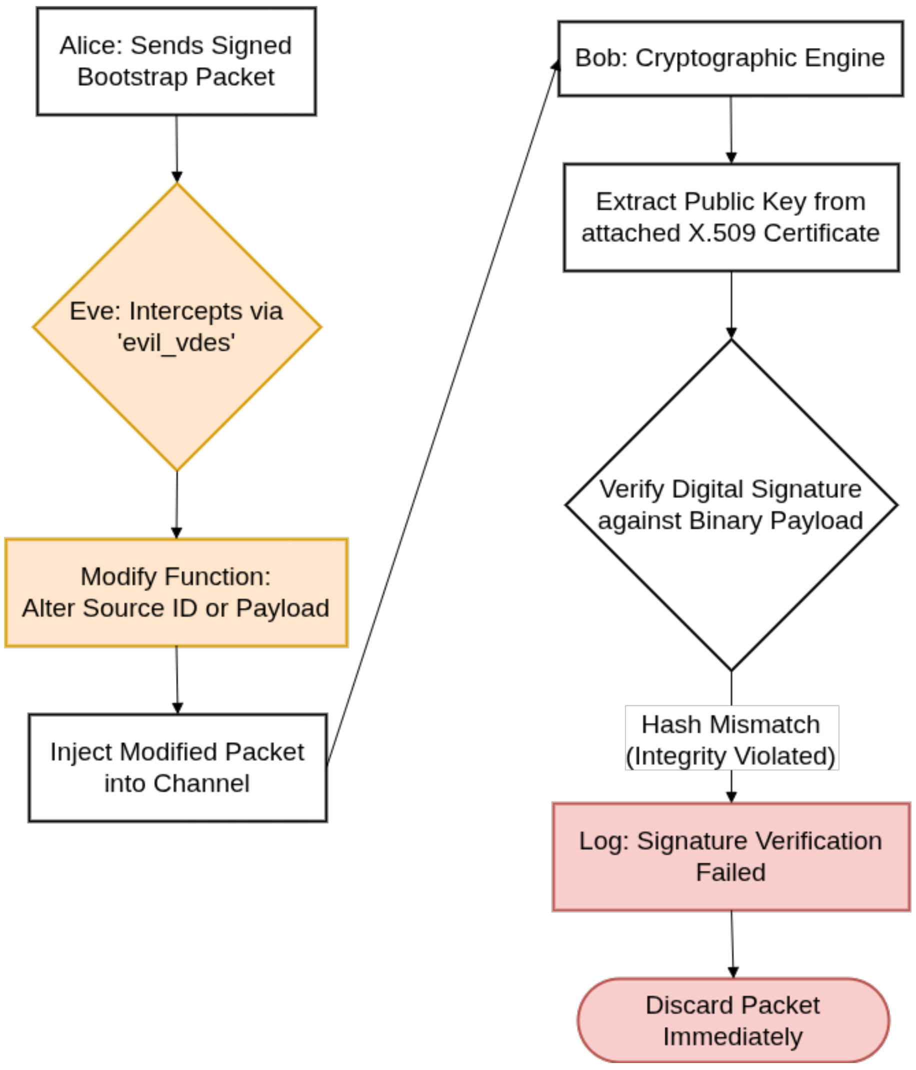

Alice: Sends Signed Bootstrap Packet
Eve: Intercepts via 'evil_vdes'
Modify Function: Alter Source ID or Payload
Inject Modified Packet into Channel
Bob: Cryptographic Engine
Extract Public Key from attached X.509 Certificate
Verify Digital Signature against Binary Payload
Hash Mismatch (Integrity Violated)
Log: Signature Verification Failed
Discard Packet Immediately

Figure 9.3: Flowchart of the Message Modification attack (Vector A). The diagram illustrates how any alteration by the attacker is detected by the receiver's cryptographic engine, leading to packet rejection.

Vector B: Replay and Temporal Linking

The attacker attempted to use the "Replay" function to re-transmit valid, previously recorded packets to force a desynchronization.

Result: The test demonstrated the efficacy of the protocol's **Temporal Chaining** mechanism. As detailed in the log analysis (Figure 9.5), the security of the sequence relies on a strict cryptographic dependency between the timestamps of consecutive messages.

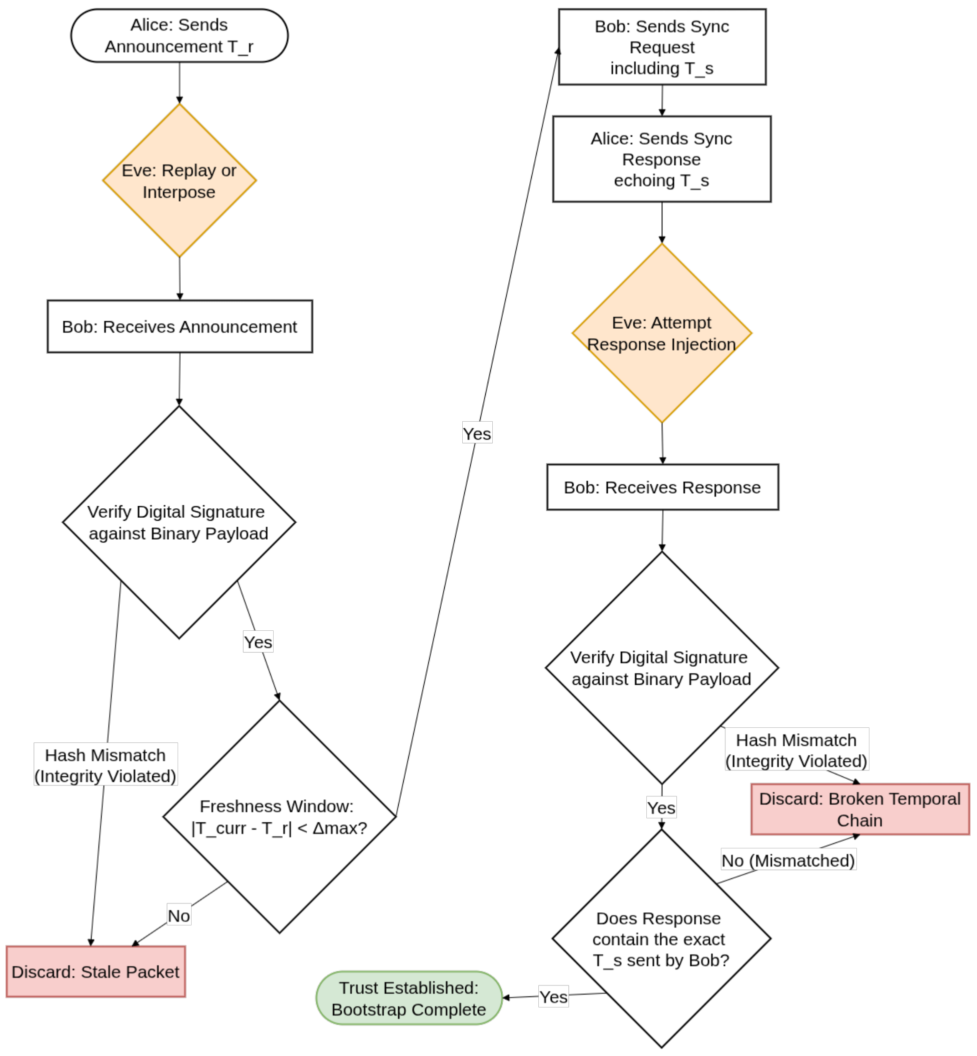
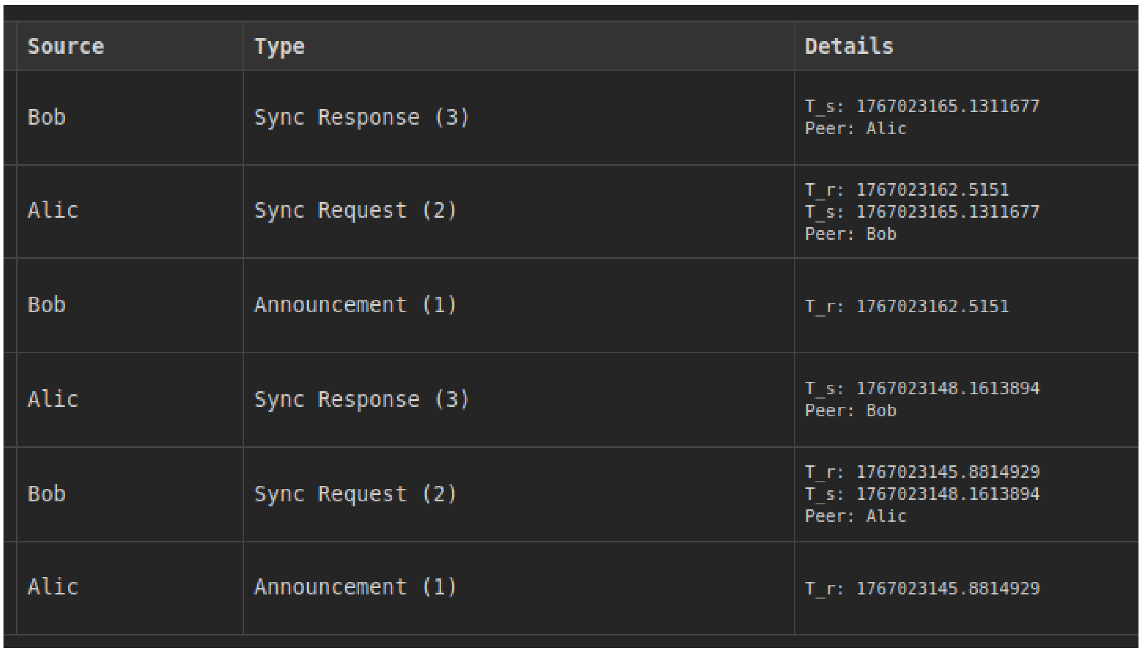

Alice: Sends Announcement T_r
Eve: Replay or Interpose
Bob: Receives Announcement
Verify Digital Signature against Binary Payload
Yes
Hash Mismatch (Integrity Violated)
Freshness Window: |T_curr - T_r| < Δmax?
No
Discard: Stale Packet
Yes
Bob: Sends Sync Request including T_s
Alice: Sends Sync Response echoing T_s
Eve: Attempt Response Injection
Bob: Receives Response
Verify Digital Signature against Binary Payload
Hash Mismatch (Integrity Violated)
Discard: Broken Temporal Chain
Yes
No (Mismatched)
Does Response contain the exact T_s sent by Bob?
Yes
Trust Established: Bootstrap Complete

Figure 9.4: Flowchart of Test Case 1 - Vector B. The diagram illustrates the dual-layer protection: digital signatures for integrity and the temporal chain (T_r and T_s) for session binding.

The "Details" column in Figure 9.5 visually confirms this "binding thread" that links the three-way handshake:



Source	Type	Details
Bob	Sync Response (3)	T_s: 1767023165.1311677 Peer: Alic
Alic	Sync Request (2)	T_r: 1767023162.5151 T_s: 1767023165.1311677 Peer: Bob
Bob	Announcement (1)	T_r: 1767023162.5151
Alic	Sync Response (3)	T_s: 1767023148.1613894 Peer: Bob
Bob	Sync Request (2)	T_r: 1767023145.8814929 T_s: 1767023148.1613894 Peer: Alic
Alic	Announcement (1)	T_r: 1767023145.8814929

Figure 9.5: Detailed view of the Temporal Chaining. The arrows implicitly indicate how the timestamp generated in one step serves as the required validation token for the next. Note how the T_r in the Sync Request matches the Announcement's time, and the T_s in the Response matches the Request's time.

1. **Link 1 (Announcement → Sync Request):**

   Alice broadcasts an announcement at time t_1 (ending in `...8814929`).

   Bob's subsequent *Sync Request* is only valid because it includes this exact timestamp in its T_r (Time Received) field. This proves Bob actually received that specific fresh announcement.

2. **Link 2 (Sync Request → Sync Response):**

   Bob's Request generates a new local timestamp t_2 (ending in `...1613894`) as its transmission time (T_s).

   Alice's final *Sync Response* is accepted by Bob only because it echoes this specific value t_2.

Any attempt to replay a packet from a previous session fails because this temporal chain is broken. As shown in the "Does Response contain the exact T_s?" decision node in Figure 9.4, replaying an old *Sync Request* would contain an obsolete T_r that does not match the current Announcement's timestamp, causing immediate rejection.

The Announcement Exception: The only packet not bound by a previous reference is the initial *Announcement* (Type 1). In this case, the system enforces a strict **Freshness Window** (see the left branch of Figure 9.4). The receiver accepted the replayed announcement only if:

$$|T_{current} - T_{packet}| < \Delta_{max}$$

Consequently, replay attacks attempted outside this narrow validity window (typically configured to a few seconds) were automatically rejected as stale, effectively neutralizing the threat [8].

9.3.2 Test Case 2: Spoofing and Integrity Violation (Operational Phase)

This experimental session evaluated the robustness of the Message Authentication Code (MAC) verification logic during the steady-state operational phase. Unlike the Bootstrap phase, which relies on asymmetric cryptography, this phase relies on the symmetric TESLA protocol and the delayed key disclosure mechanism.

The experiment was conducted using the *Evil VDES* framework to intercept, manipulate, and reinject operational packets between the synchronized stations Alice and Bob. The complete attack and defense workflow is illustrated in Figure 9.6.

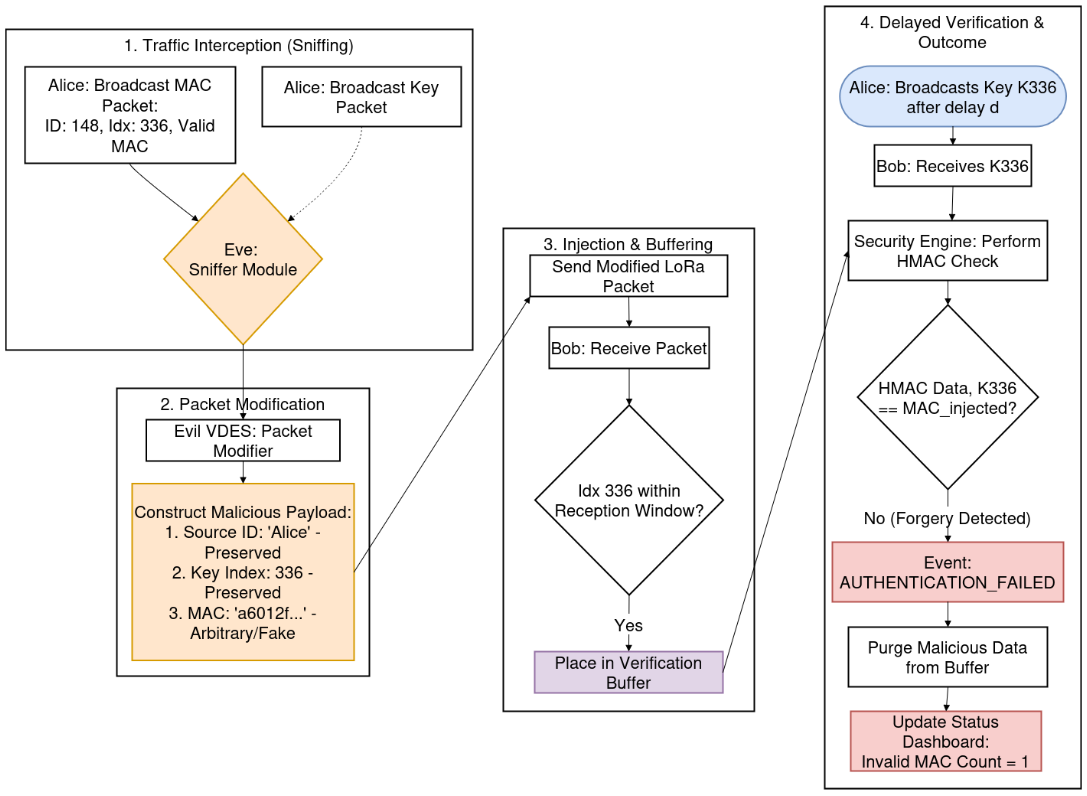

1. Traffic Interception (Sniffing)
Alice: Broadcast MAC Packet: ID: 148, Idx: 336, Valid MAC
Alice: Broadcast Key Packet
Eve: Sniffer Module
2. Packet Modification
Evil VDES: Packet Modifier
Construct Malicious Payload: 1. Source ID: 'Alice' - Preserved 2. Key Index: 336 - Preserved 3. MAC: 'a6012f...' - Arbitrary/Fake
3. Injection & Buffering
Send Modified LoRa Packet
Bob: Receive Packet
Idx 336 within Reception Window?
Yes
Place in Verification Buffer
4. Delayed Verification & Outcome
Alice: Broadcasts Key K336 after delay d
Bob: Receives K336
Security Engine: Perform HMAC Check
HMAC Data, K336 == MAC_injected?
No (Forgery Detected)
Event: AUTHENTICATION_FAILED
Purge Malicious Data from Buffer
Update Status Dashboard: Invalid MAC Count = 1

Figure 9.6: Flowchart of the Spoofing and Integrity attack (Test Case 2). The diagram highlights the four phases: sniffing, payload modification with a fake MAC, initial buffering by the receiver, and final rejection during the delayed TESLA verification check.

Traffic Interception and Analysis

First, the adversarial node passively monitored the channel to identify the current operational parameters using the Sniffer module described in Section 7.2.5. As illustrated in the interface capture previously shown in Figure 7.6, the communication pattern consists of alternating MAC packets (Type 4) and Key Packets (Type 5).

From that specific capture (Figure 7.6), the attacker identifies a valid MAC packet (ID 148) originating from 'Alice.' Crucially, the packet contains:

- `key_index`: 336 (indicating the time interval i).
- `linking_structure`: The truncated hash of the associated AIS payload.
- `mac`: The valid cryptographic tag generated by Alice using the yet-undisclosed key K_{336}.

Packet Modification and Injection

The attacker constructs a malicious payload, using the interface shown in Figure 7.8, where:

1. The `source_id` ('Alice') and `key_index` (336) are preserved to make the packet appear temporally valid.
2. The `mac` field is replaced with an arbitrary 128-bit string (e.g., beginning with `a6012f...`), as the attacker does not possess K_{336} and cannot compute a valid MAC.

Verification and Outcome

The modified packet was injected into the channel via the "Send Modified (LoRa)" command. The receiver (Bob) successfully received the packet and, noting that the `key_index` 336 fell within the acceptable reception window, placed it in the verification buffer.

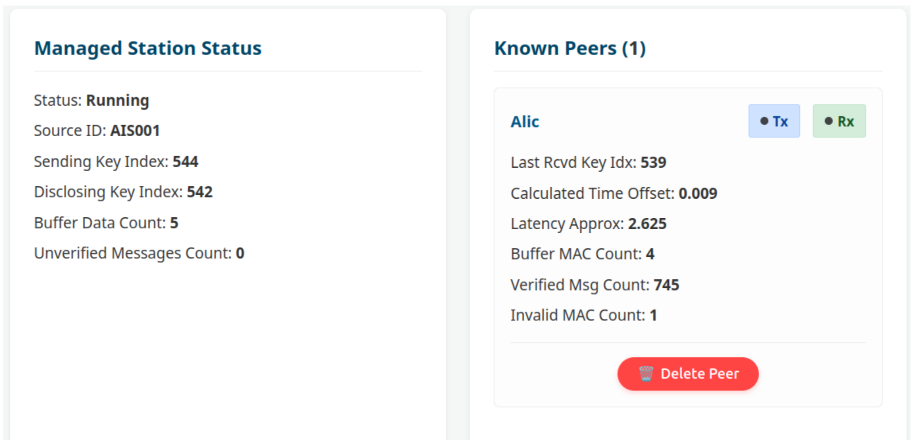
However, the TESLA security guarantee was activated upon expiration of the disclosure delay d. When Alice subsequently broadcast the key K_{336} (or a key allowing the derivation of K_{336}), Bob's security engine performed the validation check:

$$\text{HMAC}(SHA - 256(\text{Data}), K_{336}) \stackrel{?}{=} \text{MAC}_{\text{injected}}$$

Since the injected MAC was arbitrary (or calculated with an incorrect key), the equality failed. The system logged an `AUTHENTICATION_FAILED` event and immediately removed malicious data from the buffer, preventing it from reaching the application layer. This confirms that even if an attacker can inject syntactically correct packets, they cannot forge the cryptographic proof required for delayed verification.

Evidence of this successful mitigation is visible on the receiver's real-time status dashboard, shown in Figure 9.7.

The "Known Peers" panel for the sender 'Alic' explicitly registers the security event: while the `Verified Msg Count` (745) confirms the continuity of legitimate traffic, the `Invalid MAC Count` has incremented to **1**. This metric corresponds directly to the single spoofed packet injected by the adversary, definitively proving that the *Auth Interceptor* correctly segregated and discarded the forged payload upon key disclosure.



Managed Station Status

Status: **Running**
Source ID: **AIS001**
Sending Key Index: **544**
Disclosing Key Index: **542**
Buffer Data Count: **5**
Unverified Messages Count: **0**

Known Peers (1)

Alic

Last Rcvd Key Idx: **539**
Calculated Time Offset: **0.009**
Latency Approx: **2.625**
Buffer MAC Count: **4**
Verified Msg Count: **745**
Invalid MAC Count: **1**

Figure 9.7: Receiver's status dashboard following the spoofing attack. The 'Invalid MAC Count: 1' field (Right Panel) confirms that the specific malicious packet injected by Eve was detected and rejected, while 745 legitimate messages were correctly verified.

9.3.3 Test Case 3: Denial of Service and Channel Jamming

The final experimental battery measured the system's resilience against availability attacks. Using the *Evil VDES Controller*'s network attack module (illustrated in Chapter 7, Figure 7.7), two distinct strategies were executed: a logical Denial of Service via packet flooding and a physical Denial of Service via raw signal jamming.

Scenario A: Logical DoS (Replay/Flooding)

In this scenario, Eve flooded the channel with a mixture of replayed valid-header packets and arbitrary data. The impact on the receiver's internal state was immediate and observable in the station status dashboard (Figure 9.9).

Operational Degradation Analysis:

1. **Key Stream Interruption and Desynchronization:**

   The most critical evidence of the attack is the divergence between the receiver's current timeline and the sender's state. While the local `Sending Key Index` has advanced to **37**, the `Last Rcvd Key Idx` from the peer 'Alic' is stuck at

32. This gap of 5 intervals confirms that the flooding caused physical collisions, successfully blocking the reception of sequential key packets.

2. **Buffer Saturation:**

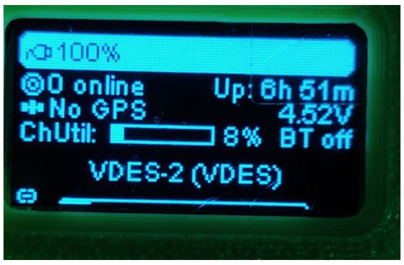
   Due to the missing keys, legitimate packets remain "orphaned" in the buffer. The dashboard shows a `Buffer MAC Count` of **63** and a `Buffer Data Count` of **10**. As detailed in the "Messages Pending Verification" table, these data packets are linked to indices (e.g., 35, 36, 37) that correspond to the missing key range, rendering them verification-pending until they eventually time out.

3. **Protocol Stress (Rejection):**

   The `Invalid MAC Count` has risen to **10**, demonstrating that the system is expending resources to process and cryptographically reject the replayed or forged packets injected by the attacker. Additionally, the `Unverified Messages Count` (**3**) indicates that some legitimate messages have already exceeded the maximum buffer retention time without being verified, resulting in data loss.

4. **Channel Saturation:**

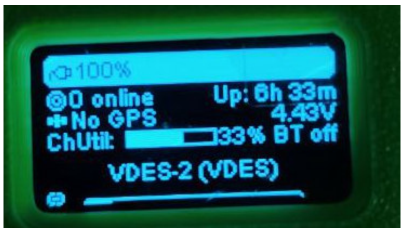
   Telemetry from the LoRa transceivers showed a Channel Utilization (`ChUtil`) increase from a baseline of $\approx 8\%$ (idle/steady state) to $\approx 33\%$ during the DoS attack. This partial saturation allows some packets to pass but drastically increases the collision rate, directly causing the key loss described above.



```
100%
0 online         Up: 6h 51m
No GPS                4.52V
ChUtil:          8% BT off
      VDES-2 (VDES)
```

(a) Baseline Operation (8%)

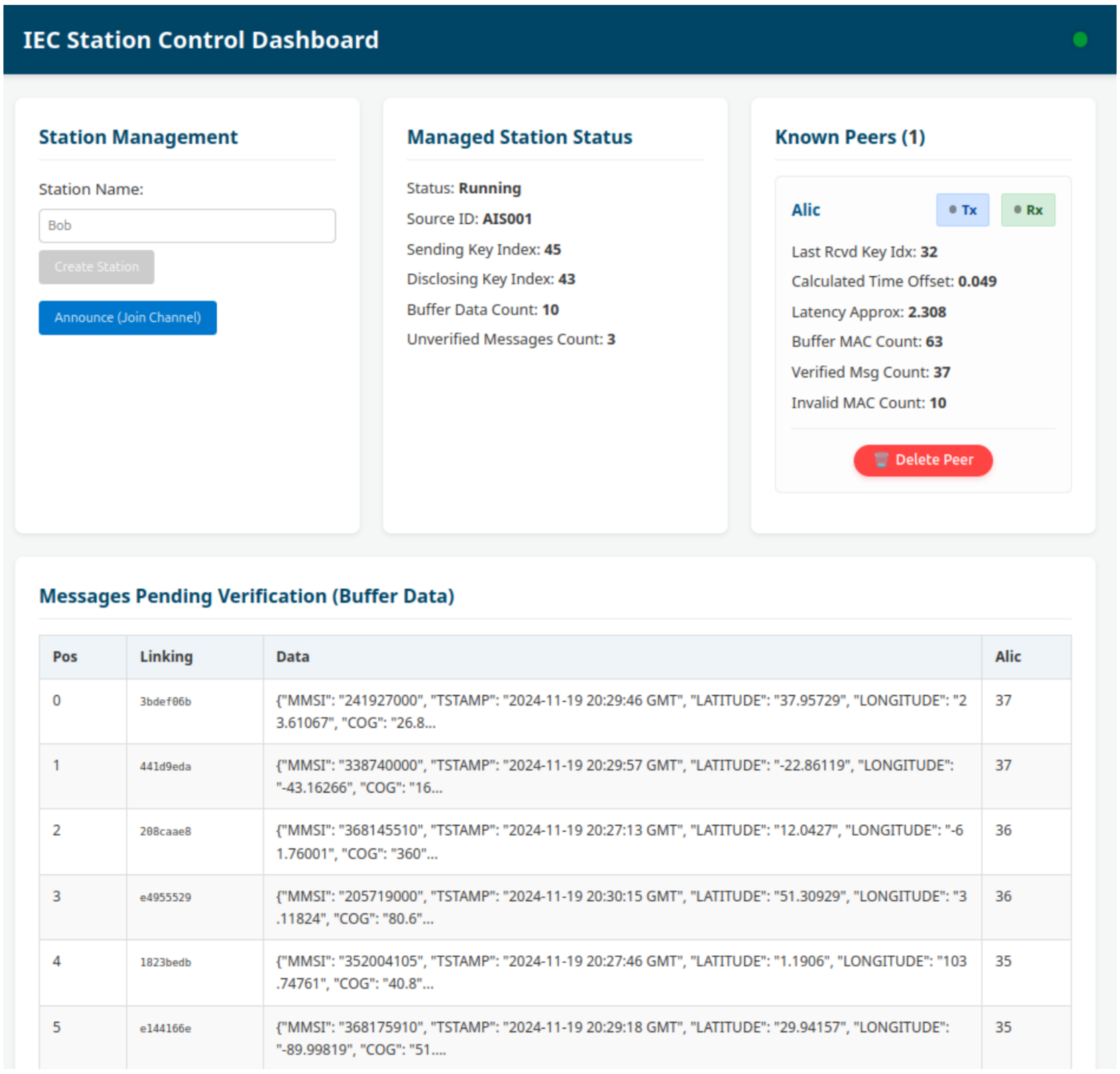
$\Rightarrow$



```
100%
0 online         Up: 6h 33m
No GPS                4.43V
ChUtil:         33% BT off
      VDES-2 (VDES)
```

(b) Under DoS Attack (33%)

Figure 9.8: Physical impact of the logical DoS attack on the LoRa radio module. The Channel Utilisation (ChUtil) spikes significantly from 8% to 33%, indicating spectrum congestion caused by the flood of replayed packets.



IEC Station Control Dashboard

Station Management

Station Name: Bob

Create Station

Announce (Join Channel)

Managed Station Status

Status: **Running**
Source ID: **AIS001**
Sending Key Index: **45**
Disclosing Key Index: **43**
Buffer Data Count: **10**
Unverified Messages Count: **3**

Known Peers (1)

Alic ● Tx ● Rx

Last Rcvd Key Idx: **32**
Calculated Time Offset: **0.049**
Latency Approx: **2.308**
Buffer MAC Count: **63**
Verified Msg Count: **37**
Invalid MAC Count: **10**

Delete Peer

Messages Pending Verification (Buffer Data)

Pos	Linking	Data	Alic
0	3bdef06b	{"MMSI": "241927000", "TSTAMP": "2024-11-19 20:29:46 GMT", "LATITUDE": "37.95729", "LONGITUDE": "23.61067", "COG": "26.8...	37
1	441d9eda	{"MMSI": "338740000", "TSTAMP": "2024-11-19 20:29:57 GMT", "LATITUDE": "-22.86119", "LONGITUDE": "-43.16266", "COG": "16...	37
2	208caae8	{"MMSI": "368145510", "TSTAMP": "2024-11-19 20:27:13 GMT", "LATITUDE": "12.0427", "LONGITUDE": "-61.76001", "COG": "360"...	36
3	e4955529	{"MMSI": "205719000", "TSTAMP": "2024-11-19 20:30:15 GMT", "LATITUDE": "51.30929", "LONGITUDE": "3.11824", "COG": "80.6"...	36
4	1823bedb	{"MMSI": "352004105", "TSTAMP": "2024-11-19 20:27:46 GMT", "LATITUDE": "1.1906", "LONGITUDE": "103.74761", "COG": "40.8"...	35
5	e144166e	{"MMSI": "368175910", "TSTAMP": "2024-11-19 20:29:18 GMT", "LATITUDE": "29.94157", "LONGITUDE": "-89.99819", "COG": "51....	35

Figure 9.9: Receiver status during DoS. Note the discrepancy between the current index (37) and the last received key (32). This causes the accumulation of 'Buffer MAC Count' (63) and 'Buffer Data Count' (10) as the receiver waits for keys that are blocked by channel collisions.

Scenario B: Physical Jamming and Recovery Dynamics

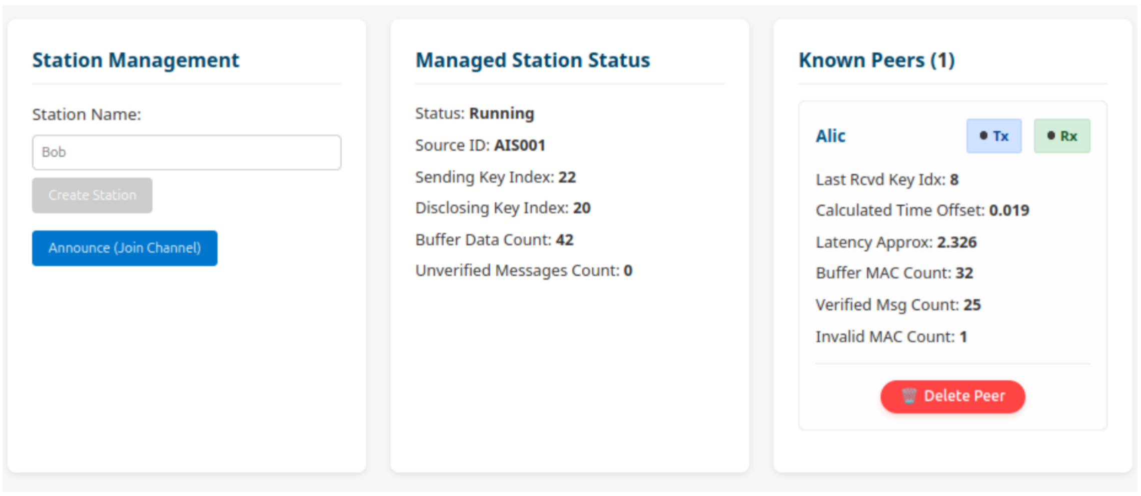
In the second scenario, Eve transmitted continuous random noise to saturate the channel. This test provided deep insight into the interaction between the physical LoRa layer (CSMA/CA) and the logical TESLA layer. The analysis is split into two phases: the active jamming phase and the transient recovery.

Phase 1: Key Starvation and Opportunistic Reception

During the attack, the noise floor prevented the reliable demodulation of long packets. As shown in Figure 9.10, the receiver enters a state of "Key Starvation":

- **Synchronization Lag:** The `Last Rcvd Key Idx` is stuck at **8**, while the station's internal time has advanced. This gap confirms that the key packets are being systematically corrupted by the jammer.

- **Opportunistic MAC Reception:** Remarkably, the dashboard shows a `Buffer MAC Count` of **32**. This indicates that despite the jamming, some authentication tags are still reaching the receiver. This is due to the **Channel Activity Detection (CAD)** and collision avoidance mechanisms inherent in the LoRa hardware. The transceiver analyzes the Channel Utilisation (`ChUtil`) and, detecting momentary micro-gaps in the jamming noise, manages to slip in the smaller, air-time-efficient MAC packets (30 bytes).
- **Result:** We have a buffer full of data and MACs, but no keys to verify them.


Station Management

Station Name: Bob

Managed Station Status

Status: **Running**
Source ID: **AIS001**
Sending Key Index: **22**
Disclosing Key Index: **20**
Buffer Data Count: **42**
Unverified Messages Count: **0**

Known Peers (1)

Alic

Last Rcvd Key Idx: **8**
Calculated Time Offset: **0.019**
Latency Approx: **2.326**
Buffer MAC Count: **32**
Verified Msg Count: **25**
Invalid MAC Count: **1**

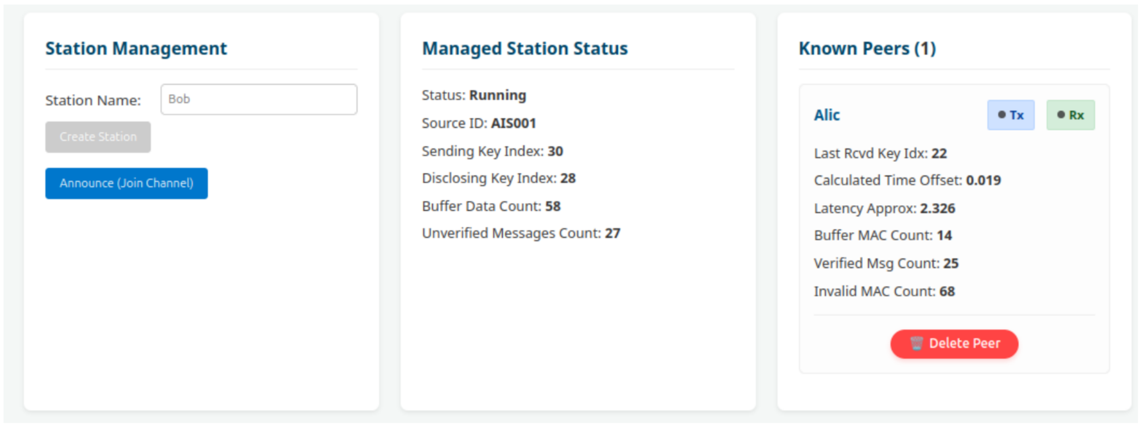
Figure 9.10: Phase 1 (During Jamming): The receiver is desynchronized (Last Key 8). However, the buffer is filling up with data (42) and MACs (32) because the LoRa physical layer manages to capture short packets during gaps in the noise floor.

Phase 2: Key Arrival and Batch Authentication

Figure 9.11 captures the moment a single key packet successfully penetrates the noise (or the jammer pauses).

- **The Recovery Event:** The `Last Rcvd Key Idx` jumps abruptly from **8** to **22**. This single successful reception carries the cryptographic material required to generate the entire chain of missing keys ($K_9 \dots K_{22}$).
- **Batch Verification:** The arrival of K_{22} triggers the TESLA verification engine, which processes the entire backlog of buffered messages in one atomic operation.
- **Outcome:** The `Buffer Data Count` does not drop to zero immediately, but shifts status. The presence of **27 Unverified Messages** indicates that while the key arrived, the latency introduced by the jamming pushed those specific

packets outside their valid security window, causing them to be discarded. However, the mechanism proves that the protocol can self-heal: a single successful packet restores the security state for the entire window of silence.



Station Management

Station Name: Bob

Managed Station Status

Status: **Running**
Source ID: **AIS001**
Sending Key Index: **30**
Disclosing Key Index: **28**
Buffer Data Count: **58**
Unverified Messages Count: **27**

Known Peers (1)

Alic ● Tx ● Rx

Last Rcvd Key Idx: **22**
Calculated Time Offset: **0.019**
Latency Approx: **2.326**
Buffer MAC Count: **14**
Verified Msg Count: **25**
Invalid MAC Count: **68**

Figure 9.11: Phase 2 (Recovery): A Key Packet (K_{22}) is finally received. This allows the system to derive the missing key chain and attempt batch authentication of the buffered packets. The rise in 'Unverified Messages' (27) reflects packets that were authenticated but deemed too old due to the jamming delay.

Conclusion on Availability

These tests empirically demonstrate the **"Fail-Secure"** behaviour of the architecture.

Under a logical attack (DoS), the cryptographic layer correctly filtered out the flood (invalid MACs) and buffered the legitimate data, although verification failed due to the lack of keys.

Under physical attack (jamming), the system simply ceased to function.

In both cases, **malicious data was not accepted as valid**, but the availability of the VDES service was successfully compromised, confirming that the physical layer remains the single point of failure in narrowband maritime communications.

Chapter 10

Conclusion

This thesis examined the problem of providing cryptographic authentication for maritime broadcast communications without disrupting the existing AIS infrastructure. Through the design, implementation, and experimental evaluation of an out-of-band architecture based on TESLA and operating over VDES, it was shown that scalable authentication can be achieved even under the strict bandwidth and interoperability constraints imposed by the VHF maritime environment.

The prototype developed in this work, together with the results obtained from the testbed and performance analysis, confirms that delayed symmetric authentication represents a viable alternative to public-key mechanisms for high-rate broadcast traffic. By separating authentication data from legacy AIS messages and sending them over the VDE-TER channel, the proposed solution preserves full backward compatibility while allowing authenticated data to be processed by VDES-capable receivers.

From a security standpoint, the most relevant outcome is the concrete reduction of the impact of AIS spoofing attacks. Current navigation and traffic management systems implicitly trust broadcast messages and cannot cryptographically verify their origin. Introducing TESLA-based authentication allows receivers to differentiate between unauthenticated and authenticated information, allowing the assigning of different trust levels to navigational data. Although malicious transmissions cannot be prevented at the physical layer, their operational effectiveness can be significantly reduced by cryptographic verification at the receiver.

An important architectural consequence of this approach is the restoration of end-to-end trust across heterogeneous communication domains. By authenticating data directly on the non-IP VHF channel, integrity and authenticity can be preserved when the same information is later forwarded to IP-based infrastructures used by shore services and secure maritime data exchange platforms. In this way, a long-standing security discontinuity between radio links and networked systems can be mitigated.

Performance evaluation highlights an unavoidable trade-off between authentication latency and transmission reliability. With the TESLA parameters selected in this study, verification occurs with a deterministic delay of approximately 6 to 9 seconds. Although this delay is higher than that of ideal public-key authentication in lossless conditions, it remains compatible with the update rates of AIS position reports. At the same time, the use of atomic authentication messages avoids fragmentation and results in more predictable behaviour under realistic VHF channel conditions, where packet loss is common.

Scalability results further indicate that symmetric authentication is well-suited to dense traffic scenarios. The limited computational cost of HMAC operations allows receivers to process data from a large number of vessels without becoming a bottleneck. This property is particularly relevant in congested maritime areas, where signature-based solutions would struggle to operate at scale.

However, several limitations of the proposed approach must be acknowledged. The architecture assumes loose time synchronisation between participants; while this is consistent with current maritime systems, significant clock drift or deliberate time manipulation could interfere with verification. Moreover, although the operational phase relies on symmetric cryptography, the initial bootstrap still depends on the conventional public-key infrastructure. As a result, the system as a whole cannot be considered fully post-quantum secure without further evolution of the bootstrap mechanism.

The experimental evaluation presented in this thesis is also constrained by the current limited availability of commercial VDES equipment. As VDES transceivers become more widely deployed and their behaviour is characterised through operational use, future studies will be able to rely on measured performance indicators such as channel latency, packet loss, and scheduling dynamics. Access to these data will make it possible to design more refined and potentially optimal authentication strategies, better aligned with the actual properties of VDES links. In particular, authentication message transmission could be further optimised, with the goal of reducing verification latency while maintaining robustness under varying channel conditions.

Beyond technical considerations, deployment challenges remain. Regulatory processes, economic factors, and the need for global interoperability are likely to slow the adoption of authenticated broadcast mechanisms. Although regional initiatives may act as early drivers, especially within the European context, partial deployment could temporarily result in environments where authenticated and unauthenticated data coexist. Addressing these issues will require coordinated efforts that involve regulators.

Bibliography

[1] IMO. *International Convention for the Safety of Life at Sea (SOLAS), 1974, as amended. Chapter V: Safety of Navigation*. International Maritime Organization, 1974.

[2] IALA. *Guideline G1192 – VDES Authentication Techniques, Edition 1.0*. International Association of Marine Aids to Navigation and Lighthouse Authorities, 06 2025. Technical Guideline.

[3] ITU-R. *Recommendation M.1371-5: Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile band*. International Telecommunication Union, 02 2014.

[4] ITU-R. *Recommendation M.2092-1: Technical characteristics for a VHF data exchange system in the VHF maritime mobile band*. International Telecommunication Union, 10 2019.

[5] IEC. Draft iec technical specification for vdes. Technical specification, International Electrotechnical Commission, 03 2017.

[6] IEC. *IEC 63173-2:2022 Maritime navigation and radiocommunication equipment and systems – Data interface – Part 2: Secure communication between ship and shore (SECOM)*. International Electrotechnical Commission, 2022. Standard.

[7] Maritime Connectivity Platform Consortium. Maritime connectivity platform (mcp) documentation. `https://maritimeconnectivity.net/`, 2024. Accessed: 2025-01-01.

[8] Adrian Perrig, Ran Canetti, J.D. Tygar, and Dawn Song. The tesla broadcast authentication protocol. RFC 4082, IETF, 06 2005.

[9] Adrian Perrig, Ran Canetti, J.D. Tygar, and Dawn Song. Efficient authentication and signing of multicast streams over lossy channels. In *Proceedings 2000 IEEE Symposium on Security and Privacy*, pages 56–73. IEEE, 2000.

[10] Gareth Wimpenny, Francisco Lazaro, Jan Safar, and Ronald Raulefs. A pragmatic approach to vdes authentication. *NAVIGATION: Journal of the Institute of Navigation*, 72:navi.681, 01 2025.

[11] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1st edition, 1996.

[12] Danny Dolev and Andrew Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 03 1983.

[13] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet x.509 public key infrastructure certificate and certificate revocation list (crl) profile. RFC 5280, IETF, 05 2008.

[14] NIST. *FIPS PUB 180-4: Secure Hash Standard (SHS)*. National Institute of Standards and Technology, 08 2015.

[15] Hugo Krawczyk, Mihir Bellare, and Ran Canetti. Hmac: Keyed-hashing for message authentication. RFC 2104, IETF, 02 1997.

[16] Peter W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pages 124–134. IEEE, 1994.

[17] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, pages 212–219, 1996.

[18] IEC. *IEC 61162-450:2018 Maritime navigation and radiocommunication equipment and systems – Digital interfaces – Part 450: Multiple talkers and multiple listeners – Ethernet interconnection*. International Electrotechnical Commission, 2018. Standard.

[19] Semtech. *SX1261/2 Long Range, Low Power, Sub-GHz Transceiver Datasheet*, 2019. Rev. 1.2.

[20] Meshtastic Project. Meshtastic firmware documentation. `https://meshtastic.org/`, 2024. Open Source LoRa Mesh Project.

[21] Docker Inc. Docker documentation. `https://docs.docker.com/`, 2024.

[22] NIST. *FIPS PUB 186-4: Digital Signature Standard (DSS)*. National Institute of Standards and Technology, 07 2013.