A PBFT-Based Consensus Approach for Secure ENC Update Protocol*

Seungbin Lee, Soowang Lee, and Jiyoon Kim†
Department of Computer Science and Engineering, Gyeongsang National University
{dltmdqls1526, id0311sw, jykim92}@gnu.ac.kr

Abstract

Electronic Navigational Chart (ENC) updates are critical for maintaining the safety of navigation in Electronic Chart Display and Information Systems (ECDIS). However, the current ENC distribution and update process faces security challenges such as data tampering, rollback attacks, and delayed propagation of critical updates. This study proposes a Practical Byzantine Fault Tolerance (PBFT)-based secure protocol to ensure integrity, authenticity, and timely delivery of ENC updates. The proposed protocol integrates a permissioned blockchain among Regional ENC Coordinating Centers (RENCs) and Hydrographic Offices (HOs), anchoring update metadata and cryptographic hashes in an immutable ledger while preserving the existing S-63/S-101 distribution pipeline. Especially, consensus-based decision making via PBFT requires cross-validation of multiple nodes to each update. It can practically suppress forgery and injection attacks on single nodes or distribution paths. Motivated by this, we propose ENC update protocols for ECDIS system and verify its security through Scyther, a formal verification tool.

Keyword: ECDIS, ENC Update, PBFT, Protocol, Formal Verification

1 Introduction

Electronic Chart Display and Information Systems (ECDIS) is a digital replacement for paper charts. While paper charts have traditionally been used, ECDIS was developed to address their increasing inconveniences and numerous challenges. ECDIS integrates Electronic Navigational Charts (ENCs) with real-time positional and environmental data. This enables navigators to make informed decisions that avoid maritime hazards and comply with international maritime regulations (International Hydrographic Organization [IHO], 2020; IHO, 2022). Because the marine environment is highly dynamic, ENCs require continuous updates. These updates include newly identified hazards, relocated buoys, modified traffic separation schemes, and hydrographic survey results. Such updates are essential for operational efficiency, the safety of marine life, and the protection of the marine environment (IHO, 2023a).

ENC updates are currently distributed through Hydrographic Offices (HOs), Regional ENC Coordination Centers (RENCs), and a network of authorized distribution agents, using standards defined by the IHO (IHO, 2023b; IHO, 2023c). Updates are provided as encrypted and digitally signed files, adhering to the S-57/S-63 framework as well as the emerging S-100/S-101 standards

* Proceedings of the 9th International Conference on Mobile Internet Security (MobiSec'25), Article No. 58, December 16-18, 2025, Sapporo , Japan. © The copyright of this paper remains with the author(s).

† Corresponding author

(IHO, 2020; IHO, 2022). The S-63 method ensures that only authorized ECDIS units can decrypt the data, and any tampering during transmission can be detected through signature verification (IHO, 2023b). However, the current system highly depends on a centralized trust anchor and a batch-based distribution schedule. In recent years, the problems associated with this dependency have become increasingly serious. Detailed security and operational concerns are discussed below (IHO, 2023a; International Chamber of Shipping & BIMCO, 2020).

The system has a centralized structure. This makes it an attractive target for attackers as a single point of failure. Attackers can inject forged ENC files into the supply chain by launching man-in-the-middle attacks on distribution servers or communication channels (IHO, 2023a; Hrcak, 2024). Additionally, the reintroduction of outdated ENC versions lacking critical safety updates could result in hazardous navigation decisions (IHO, 2023a). The existing cryptographic key management model is post-counteraction and often delayed. Revocation notices are often delayed, sometimes taking days or even weeks after a key infringement occurs. During this interval, maliciously signed and unauthorized updates may continue to be accepted (IHO, 2023a; IHO, 2023b). Furthermore, delays that jeopardize safety may arise during emergency deployments. This is because the deployment workflow is not optimized for Temporary and Provisional (T&P) notices or high-priority emergency deployments(IHO, 2023a; Riviera Maritime Media, 2022).

To address vulnerabilities in the current system, a secure distributed update verification mechanism is necessary. This mechanism does not require replacing existing IHO-compliant distribution channels, providing a complementary approach instead. It ensures global consistency, immediate finality of approvals, and rapid dissemination of critical updates (International Chamber of Shipping & BIMCO, 2020; Hrcak, 2024). Practical Byzantine Fault Tolerance (PBFT) offers a robust solution in this context. PBFT is designed for permissioned environments with a fixed set of known validators (Castro & Liskov, 1999; Yin et al., 2019). It ensures deterministic finality while maintaining resilience against up to one-third of Byzantine faults. These features make PBFT especially suitable for ENC updates, where the integrity of updates and the irreversible ordering of finalized data are critical (Li et al., 2020; Zhang et al., 2021).

This paper proposes a PBFT-based consensus protocol for the secure and efficient distribution of ENC updates in ECDIS. It was designed to maintain compatibility with existing IHO standards while enhancing security and reliability. Update metadata is recorded in an immutable ledger through a consensus-algorithm blockchain network established between RENCs and HOs. A key aspect is the separation of the metadata consensus layer from the existing file distribution layer. The consensus layer provides tamper-proof records and rollback prevention through metadata consensus.

The contributions of this paper are as follows:

- Identifying security vulnerabilities in ECIDS update pipeline systems, such as malicious data injection, rollback attack, and delayed key revocation.
- We designed a protocol that can guarantee data integrity through a consensus layer while being compatible with existing IHO standards.
- Verification is conducted to prove cyber threats and security properties through a formal verification tool, Scyther.

The structure of this paper is as follows:

In section 2, we reviewed ENC deploy environment and IHO standard. Additionally, we analyze the recent research on the consensus algorithm, PBFT. Section 3 analyzes the security threats that can arise from ENC updates. In Section 4, we enhanced ENC update protocol with PBFT algorithm. Section 5 demonstrates that the security of the proposed consensus protocol is verified using the formal verification tool Scyther. Section 6 describes the conclusion of this paper and future research directions.

2 Related Work

2.1 2.1 IHO ENC Standards: S-57, S-63, and S-100/S-101

The IHO defines the technical and operational standards for the production, protection, and distribution of ENCs within the ECDIS ecosystem. These standards establish the fundamental data models and the security and interoperability requirements for HOs, RENCs, equipment manufacturers and end users(IHO, 2020, 2022, 2023a).

S-57 is the initial digital hydrographic data transfer standard, adopted in 1990. A logical data structure and a feature catalogue for encoding hydrographic objects are specified by the standard. Examples include coastlines, depth contours, aids to navigation, and traffic separation schemes. S-57 is a data model and exchange format, and does not address protection, encryption, or distribution security(IHO, 2020). Therefore, the S-63 data protection scheme was developed to safeguard ENC distribution. The definition of S-63 is as follows:

(i) Data access to licensed ECDIS units is restricted by symmetric encryption (blowfish).
(ii) File authenticity is verified and tampering is detected through the use of digital signatures (RSA).
(iii) Public Key Infrastructure (PKI) is managed by the IHO, with an IHO Certificate Authority and subordinate authorities for HOs and RENCs.
(iv) ENC usage is bound to a specific ECDIS system serial number via a permit file.
(v) Processes for key issuance, distribution, and revocation.

S-63 authenticates the origin of ENC files and ensures their integrity during transmission(IHO, 2023b). However, a problem arises in which outdated or reissued cells are redistributed even when a valid signature exists. Additionally, there is no mechanism to coordinate update sequences across multiple deployment points. Key revocation and rotation rely on manual, batch-based procedures, which may delay the mitigation of key compromises(IHO, 2023a). The S-100 hydrological data model was introduced by the IHO to address a wide range of hydrological data requirements. This complies with the ISO/TC 211 framework, which supports not only bathymetric grids, tidal data, and marine environmental datasets, but also a wide range of product types(IHO, 2022). S-100 offers:

(i) A GML-based and extensible data model.
(ii) Common metadata structures for geospatial datasets.
(iii) Support for time-dependent and 3D/4D datasets.
(iv) An interoperability framework for displaying multiple product types simultaneously.

Within the S-100, S-101 defines the next generation of ENCs. It retains the navigational focus of S-57 while enabling more detailed feature attribution and rule representation. The accuracy of information, as well as the update mechanism, has also been improved(IHO, 2022). However, the S-101 security model predominantly inherits the framework of S-63(IHO, 2023b). Although the data model provides greater flexibility, trust and validation mechanisms continue to be centralized and file-centric. The limitations of S-63 and S-101 security are as follows:

(i) Global update ordering is not guaranteed, and rollback prevention mechanisms are absent.
(ii) Even when updates are approved, immediate and definitive finality is not guaranteed. There exists a risk that approved updates may be reverted or overwritten.
(iii) Lack of mechanisms for rapid, synchronized propagation of key lifecycle events.
(iv) Immutable and auditable logging for update approval and authorization events is absent.

These limitations necessitate consideration of a complementary consensus layer that is compatible with existing standards and ensures data integrity, currency, and portability.

2.2 Literature Review

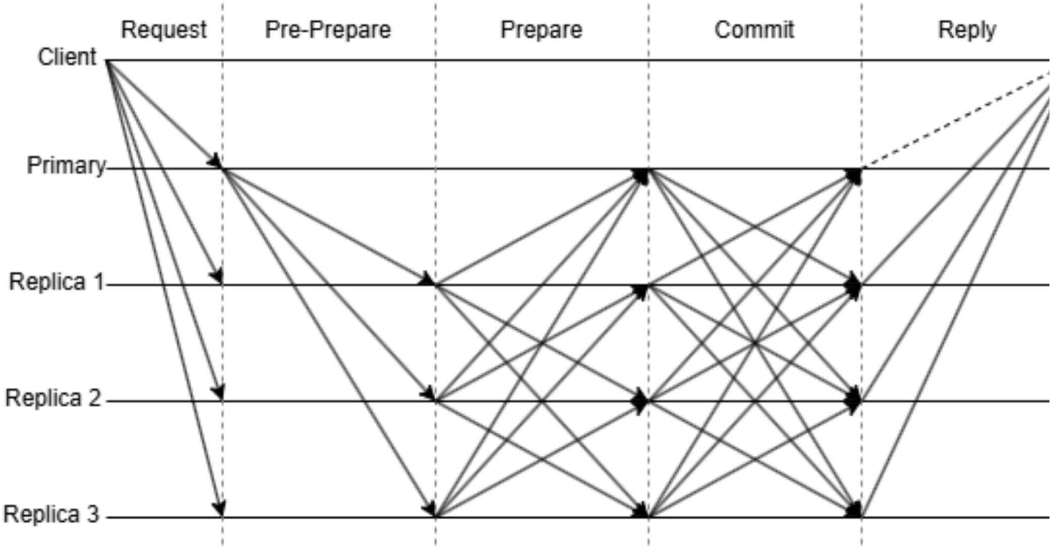


Figure : Pract**1**cal Byzantine Fault Tolerance Procedure

Practical Byzantine Fault Tolerance (PBFT), introduced by Castro and Liskov (1999), has been widely studied and implemented in permissioned blockchain frameworks such as Hyperledger Fabric, BFT-SMaRt, and Tendermint. PBFT achieves consensus among a fixed set of validators, tolerating up to f Byzantine faults in a group of 3f + 1 nodes. Its immediate finality, deterministic operation, and relatively low latency in small to medium-sized networks make it suitable for applications where reversals are unacceptable and strong consistency is critical (Yin et al., 2019). Figure 1 shows the entire PBFT procedure. Applications of PBFT and its variants have been explored across several domains that face similar integrity and trust challenges to those in maritime navigation.

These studies demonstrate PBFT's adaptability to diverse operational environments, from highly dynamic IoT networks to safety-critical transportation and logistics systems. However, none directly address the specific requirements of Electronic Navigational Chart (ENC) updates in the ECDIS ecosystem. Unlike the above use cases, ENC distribution operates in a globally governed, permissioned network of hydrographic authorities, requiring strict version dependency enforcement, anti-rollback guarantees, and synchronized key lifecycle management. Our proposed PBFT-based consensus protocol leverages the consensus-layer design principles from these studies—such as leader randomization, reputation weighting, and metadata/application separation—while tailoring them to the unique operational and regulatory constraints of maritime navigation.

R-PBFT (Kumar et al., 2023) augments standard PBFT with a reputation-weighted voting mechanism, where node trust scores are computed via logistic regression over behavioral metrics. Designed for Internet of Vehicles (IoV) environments, it targets high mobility and potential Sybil/Eclipse/impersonation threats. Experiments on Raspberry Pi 4 and NVIDIA Jetson Nano testbeds show significant improvements over baseline PBFT—about 50% reduction in consensus latency, 50% increase in throughput, and 57% lower congestion, with improved fairness. The limitation is that reputation computation and trust bootstrapping are tuned for dynamic, loosely governed networks, whereas ENC distribution operates in a stable, institutionally controlled validator set. For our ENC protocol, the batching and timeout optimizations demonstrated here are relevant for satellite-latency maritime links, while reputation weighting would be optional rather than core.

Hegde et al. (2023) propose a PBFT variant incorporating the EigenTrust reputation model to select trustworthy consensus nodes and a Verifiable Random Function (VRF) to randomly assign leaders, thereby reducing the predictability of leadership and mitigating targeted attacks. Evaluations in simulated medical IoT networks show latency reduction and throughput gains compared to standard PBFT. Its limitation is that leader-targeting mitigation is only partially explored in adversarial field conditions. For our proposed protocol, the leader randomization concept directly applies to our RENC/HO validator set to harden against targeted disruption, while the overall "trusted node" filtering aligns with the controlled nature of hydrographic governance.

Wu et al. (2025) propose DBPBFT, which separates responsibilities between two blockchains—one for consensus and one for application data—while employing a hierarchical PBFT process within each. The architecture aims to scale large IoT deployments without compromising safety. The limitation is the operational overhead of maintaining cross-chain consistency and verification. Our ENC protocol shares the "dual-path" philosophy: the consensus ledger anchors metadata (hashes, versions, key events), while existing S-63/S-101 channels carry the actual ENC files, thereby minimizing disruption to the operational distribution pipeline.

Xu et al. (2023) introduce ABC-GSPBFT, which integrates group scoring of nodes and Artificial Bee Colony metaheuristics to preselect reliable validators and simplify block proposal phases. Applied to aviation operational data sharing, it reduces delay and overhead while ensuring security. Its limitation is reliance on aviation-specific assumptions about connectivity and data continuity. For our ENC protocol, the shared aviation/maritime need for rapid, authoritative dissemination of safety-critical notices is clear; however, unlike flight data, ENC updates must enforce strict dependency closure (e.g., edition/update number chains) before application.

Liu and Zhu (2024) propose the AP-PBFT algorithm to address a limitation of classical PBFT—in which consensus is decided by a single leader with other nodes merely validating. AP-PBFT allows nodes to express preferences over multiple proposals, with a VRF used to select both the primary proposer and consensus nodes randomly. Consensus proceeds by having nodes independently vote on proposals, running a "consensus output protocol" locally, and then letting the leader aggregate these into a final decision. To promote honest behavior, the authors incorporate an incentive mechanism and evolutionary game model, penalizing dishonest votes and rewarding compliant behavior. Through simulation, AP-PBFT demonstrates improved scalability and throughput, and better adaptability to dynamic node membership, while preserving consistency, validity, and termination properties. However, its preference-aggregation model is tailored toward multi-valued consensus scenarios (e.g., DAO decisions), not settings requiring a single correct outcome, like ENC updates. In our proposed PBFT-based ENC update protocol, we can borrow the VRF-based leader randomization to resist targeted attacks, but we retain a single-value, deterministic consensus process to ensure strict edition/update correctness and anti-rollback guarantees.

Zhang et al. (2024) introduce NR-PBFT, which adapts PBFT for marine fishery cold-chain tracking, aiming to improve throughput, reduce delay, and cut communication overhead under maritime communication constraints. While relevant in considering sea-based link limitations, it does not address the formal dependency and versioning constraints of standardized ENC data. For our ENC protocol, NR-PBFT's assumption of a permissioned, maritime-context validator set aligns closely, but our focus extends to embedding edition/update numbers, dependency checks, and key lifecycle events into the consensus layer.

Despite these successful applications, there is a notable research gap in applying PBFT to the maritime sector, and specifically to the ECDIS ENC update process. Existing maritime cybersecurity studies have focused primarily on protecting shipboard networks, securing Automatic Identification

System (AIS) transmissions, and preventing GPS spoofing (Balduzzi et al., 2014; Tam & Jones, 2018) Very few have addressed the integrity and freshness of ENC updates using distributed consensus approaches. Moreover, current ENC protection measures operate at the file level, leaving the broader update sequencing, dependency management, and global consistency unaddressed. Our proposed PBFT-based secure protocol seeks to fill this gap by introducing a consensus layer among trusted maritime authorities, thereby ensuring that only consensus-approved updates are recognized and applied by ECDIS units worldwide. This approach not only mitigates key vulnerabilities but also aligns with ongoing trends toward digitalization and secure data sharing in the maritime industry.

3 Security Threats in ENC Updates

One of the most critical Security vulnerabilities in the ENC update process is malicious data injection and data impersonation. If a distribution server, relay node, or network is compromised, an attacker can insert a forged ENC file or forge the Permit.txt file with the update process. Studies have shown that the ECDIS system is interconnected with the external interface, such as radar, GPS, Internet, and LAN, which makes them vulnerable to supply-chain and transmission-path attacks (Svilicic et al., 2019). The S-63 standard ensures reliability that offers file-level encryption and a digital signature. however, data cannot be blocked during transmission as an unsigned file. These vulnerabilities lead to incorrect chart information, posing a direct threat to navigational safety.

The other vulnerabilities are rollback and downgrade attacks. In this attack scenario, an outdated ENC version or an update is replaced that includes essential fixes, either intentionally or through an operational error. The current system only verifies the authenticity of files and does not verify their location information in the global update sequence. This does not fundamentally prevent the approval of validly signed but invalid ENC updates. This vulnerability is especially dangerous when Temporary and Preliminary (T&P) Notices or other urgent updates are involved. According to research, update delay and operational errors increase navigational risk(Kayişoğlu et al., 2024).

Another significant security concern, key life-cycle compromise and delayed revocation. If the private key of an IHO or Regional ENC Coordinating Center(RENC) is exposed, an adversary could produce a validly signed but malicious updates. In the current permit-based key management systems, the propagation of the key revocation message across the global fleets or vessels can even take a few days or weeks. In large-scale PKI environments, similar delays in revocation have been observed, where compromised certificates remain trusted for long-terms (Kim et al., 2018; Zhu et al., 2016).

The update process is exposed trasmit channel or the mobility media through malware infection. The ENC update can offer USB drive, DVD, or a bridge connection PC. These external devices can be potential vectors for spyware, ransomware, and other malicious software. In addition, the directly connected ECDIS device in an external network can be target of Denial-of-Service(DoS) attacks, which can delay or block critical updates(Svilicic et al., 2019; Bothur, 2017).

Incomplete or inconsistent application of updates presents another risk. This can occur when updates with dependencies on other ENC cells are applied in isolation, or when manual data entry and local procedures introduce errors. Discrepancies may also arise from differences between onshore SENC (System ENC) conversion processes and onboard applications, leading to mismatched chart states that can mislead navigators (Kayişoğlu et al., 2024).

The network architecture itself that supports ENC updates may be the source of vulnerabilities. If the initial entry point is compromised due to insufficient separation between ECDIS, back-of-bridge IT systems, and external networks, the security threat can be for lateral movement. This further increases the attack surface when using third-party applications or personal devices with the update process(Bothur, 2017; Oruc & Aydin, 2025).

Using Insecure or Informal ENC data leads to critical danger. Loading an unencrypted S-57 standard file or enc data from unofficial sources compromises the reliability and integrity of the supply chain, at risk of supply chain impersonation or injection attacks(Bothur, 2017).

Finally, human and procedural factors are identified as causes of continuous vulnerabilities. Factors such as adherence to established update procedures, insufficient training and accountability among navigators, inadequate record management, and inconsistent auditing practices can undermine the benefits of technological safeguards. Inconsistencies between ECDIS configurations or the inability to apply rapid updates have been identified as operational challenges in several maritime cybersecurity studies(Kayişoğlu et al., 2024).

In the ENC update process, the major security threats include malicious data injection, rollback and downgrade attacks, and delayed revocation issues arising during the key lifecycle. These vulnerabilities are not merely technical flaws but pose a direct risk to navigational safety, thereby necessitating more sophisticated security enhancements. To mitigate these threats, a consensus-algorithm-based verification procedure can be introduced to detect and prevent data forgery and tampering. the combination of randomly generated nonces and sequence numbers can ensure both the integrity and freshness of updates. Such structural enhancements not only prevent malicious data injections or the reapplication of outdated versions but also provide mechanisms to address potential vulnerabilities in key management. This approach strengthens the resilience and reliability of the ENC update system, thereby substantially enhancing security across the navigation process.

4 Proposed Protocol

4.1 Consensus Procedure

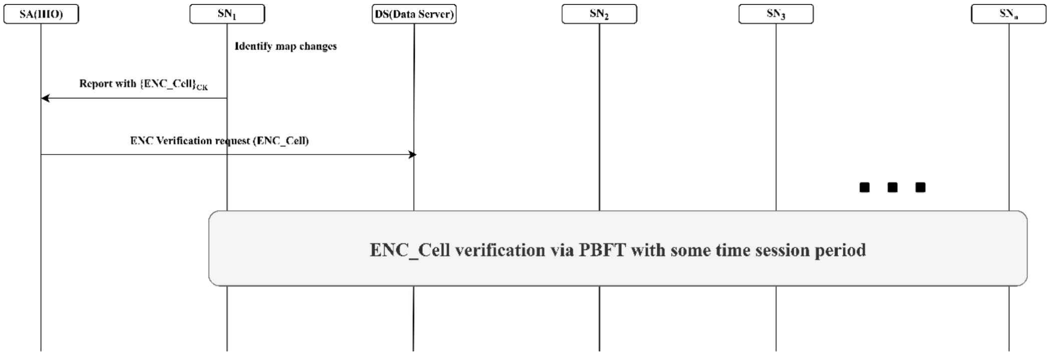

SA(IHO)
SN_1
DS(Data Server)
SN_2
SN_3
SN_n
Identify map changes
Report with $\{ENC\_Cell\}_{CK}$
ENC Verification request (ENC_Cell)
ENC_Cell verification via PBFT with some time session period

Figure 2: Proposed Consensus Procedure for ENC Update

The proposed consensus procedure for ENC update is shown in Figure 2. The procedure is described in detail below.

(1) Ship Node (SN) identified ENC data changes.

(2) SN_1 report to SA(IHO) with ENC_{Cell} changed with identified information.

(3) SA(IHO) received changed ENC_{Cell} and request ENC verification to DS.

(4) DS performs ENC_{Cell} verification using the PBFT algorithm for SNs associated with a specific time session within a given period

4.2 Proposed PBFT Procedure

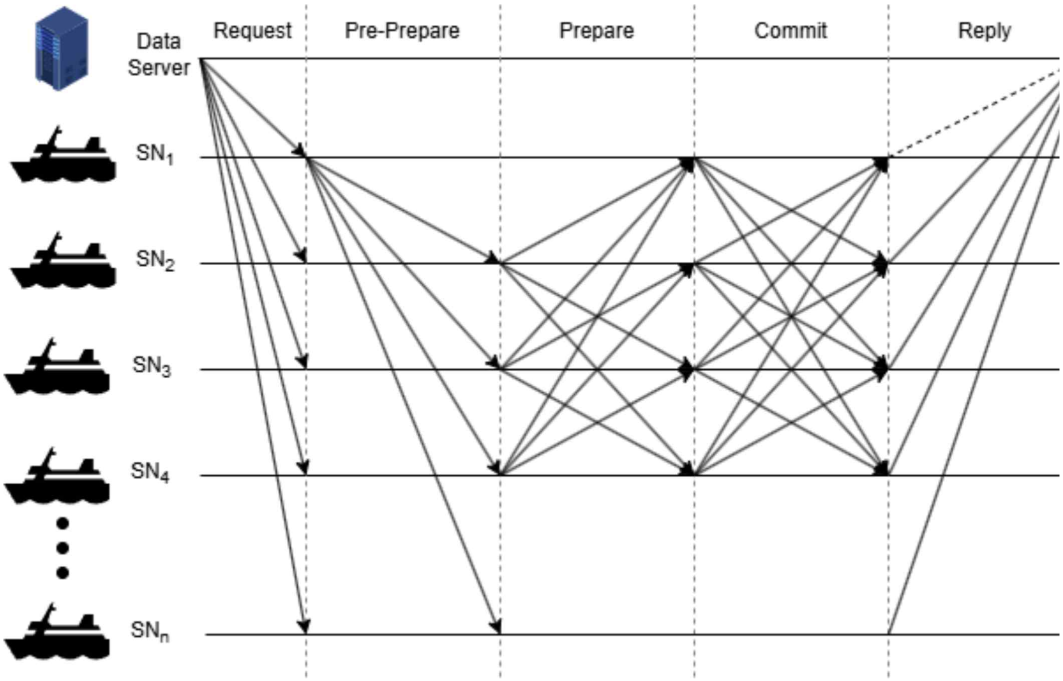


Figure : PBFT3 procedures in the proposed decision-making process.

The proposed ENC PBFT procedure is shown in Figure 3, and the protocol consists of five sequential phases: Request, Pre-Prepare, Prepare, Commit, and Reply.

Request Phase: The Data Server initiates the process by sending a request message to the primary SN_1.

Pre-Prepare Phase: Upon receiving the request, the primary SN_1 assigns a sequence number to it and multicasts a pre-prepare message to all other SNs (SN_2 … SN_n) participating in the same session. This message includes the digest of the request and the assigned sequence number, establishing the execution order.

Prepare Phase: Each ShipNode verifies the validity of the pre-prepare message and then broadcasts a prepare message to all other SNs. This step ensures that all non-faulty SNs agree on the same sequence number and request digest before proceeding to commitment.

Commit Phase: After receiving a sufficient number of consistent prepare messages (at least $2f + 1$, where f denotes the maximum number of faulty ShipNodes), each SN sends a commit message to all others. This phase guarantees that a majority of trustworthy SN have reached consensus, thereby maintaining system consistency even in the presence of Byzantine faults.

Reply Phase: Once a SN collects enough commit messages ($2f + 1$), it executes the requested operation and sends a reply message back to the Data Server. The Data Server accepts the result only after receiving matching replies from at least $f + 1$ distinct SNs, confirming that consensus for the session has been successfully achieved.

4.3 Update Procedure

The proposed ENC update procedure for ENC update is shown in Figure 4, and detailed explanations follow.

(1) SA transmits $\mathrm{M_{Key}}$ and $\mathrm{ID_M}$ with Secure channel. This procedure is out of band

(2) SN stored $\mathrm{M_{Key}}$. Afther that, The Identify $\mathrm{ID_{HW}}$, sequence number Seq# and random nonce $\mathrm{n_1}$ are generated by SN. At this time SN compute message authentication code $\mathrm{HM_1}$. The SN transmits $\mathrm{ID_{HW}}$, $\mathrm{ID_M}$, $\mathrm{n_1}$, Seq#, and $\mathrm{HM_1}$ to DS.

(3) DS verify $\mathrm{HM_1}$ with $\mathrm{M_{Key}}$. The DS generates random nonce $\mathrm{n_2}$ and enc cell info $\mathrm{ENC_{cell}}$, $\mathrm{ENC_{file}}$. And also SN compute cell key CK with $\mathrm{M_{key}}$, $\mathrm{ID_M}$, $\mathrm{ID_{HW}}$, $\mathrm{ID_{DS}}$, $\mathrm{n_1}$, and $\mathrm{n_2}$. The DS transmits to the SN a message consisting of its identifier $\mathrm{ID_{DS}}$, two nonces $\mathrm{n_1}$ and $\mathrm{n_2}$, the encrypted cell and file under the cell key CK, a digital signature over these elements using the DS's private key $\mathrm{SK_{DS}}$, together with the DS's public key, which is certified by the SA.

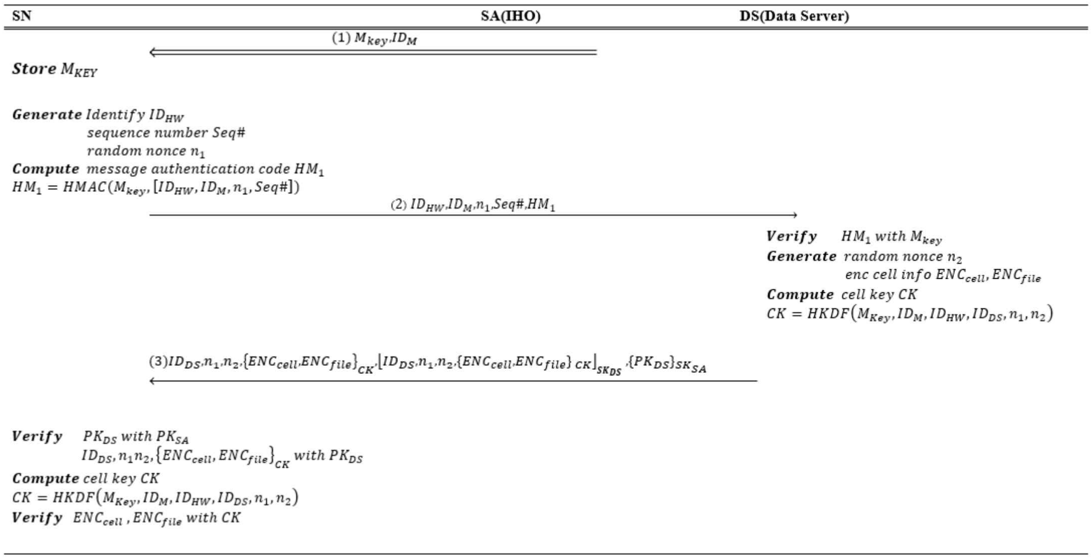

SN
SA(IHO)
DS(Data Server)
(1) M_{key}, ID_M
Store M_{KEY}
Generate Identify ID_{HW}
sequence number Seq#
random nonce n_1
Compute message authentication code HM_1
$HM_1 = HMAC(M_{key}, [ID_{HW}, ID_M, n_1, Seq\#])$
(2) $ID_{HW}, ID_M, n_1, Seq\#, HM_1$
Verify HM_1 with M_{key}
Generate random nonce n_2
enc cell info ENC_{cell}, ENC_{file}
Compute cell key CK
$CK = HKDF(M_{Key}, ID_M, ID_{HW}, ID_{DS}, n_1, n_2)$
(3)$ID_{DS}, n_1, n_2, \{ENC_{cell}, ENC_{file}\}_{CK}, \lfloor ID_{DS}, n_1, n_2, \{ENC_{cell}, ENC_{file}\}\ CK \rfloor_{SK_{DS}}, \{PK_{DS}\}SK_{SA}$
Verify PK_{DS} with PK_{SA}
$ID_{DS}, n_1 n_2, \{ENC_{cell}, ENC_{file}\}_{CK}$ with PK_{DS}
Compute cell key CK
$CK = HKDF(M_{Key}, ID_M, ID_{HW}, ID_{DS}, n_1, n_2)$
Verify ENC_{cell} , ENC_{file} with CK

Figure **:** Propo4sed Secure ENC Update Protocol

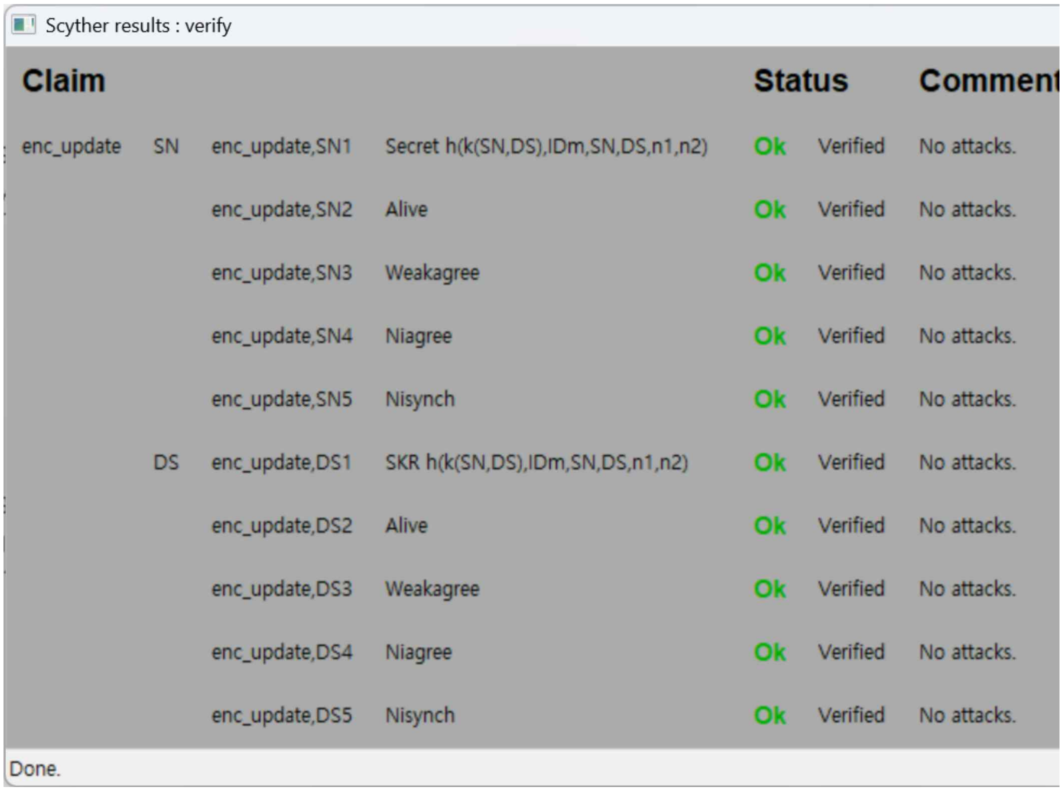
(4) SN verify $\mathrm{PK_{DS}}$ with $\mathrm{PK_{SA}}$. And $\mathrm{ID_{DS}}$, $\mathrm{n_1}$, $\mathrm{n_2}$ { $\mathrm{ENC_{CEll}}$, $\mathrm{ENC_{file}}$} $_{\mathrm{CK}}$ with $\mathrm{PK_{DS}}$. After that, SN Compute cell key CK with using $\mathrm{M_{Key}}$, $\mathrm{ID_M}$, $\mathrm{ID_{HW}}$, $\mathrm{ID_{DS}}$, $\mathrm{n_1}$, $\mathrm{n_2}$. and verify $\mathrm{ENC_{cell}}$, $\mathrm{ENC_{file}}$ with CK

5 Formal Verification

In this section, our proposed consensus-based protocol for ENC Update is verified for its security using Scyther (Cremers, 2008), an automated security verification tool. Scyther is a widely used and representative tool in the field of formal verification. It is a well-known tool whose reliability and utility have been proven through sufficient verification cases for various protocols, and performed verification based on the Dolev-Yao model. This model automatically simulates Most operations possible on a network(hijacking, Fabrication, replay attack). This tool is used for automated verification and mathematical assurance, and its usefulness is guaranteed.

Before using Scyther, we analyze the proposed protocol and interpret it to SPDL. Protocols interpreted with SPDL are modeled as Entities, each with its own participant/roles. After defining each entity's roles, verify it against security requirements such as integrity, confidentiality, mutual authentication, secure key exchange. Each security requirement is analyzed using a query defined in the claim() format. This allows verification of compliance with the security requirements.

Figure 5 shows the verification results. For the protocol we propose, we progressed formal verification with Scyther, targeting the SN and DS, which are direct participants in communication. As shown in Figure 5, SN and DS derived results "OK" for core security properties such as Alive, Weakagree, Niagree, and Nisynch. Alive ensures that each session involves real participants and that the message exchange progresses normally. This prevents protocol execution from being compromised by fake participants or incomplete sessions. Weakagree guarantees a minimum level of mutual awareness. When one participant recognizes another in a session, the other participant is also confirmed to be participating in the same session. Additionally, Niagree ensured each participant agreed to exchange data in the session. Even if multiple executions occur, data integrity is preserved, and it can offer resilience against replay attacks. Nisynch ensured that each participant's session was synchronized.



Claim				Status		Comment
enc_update	SN	enc_update,SN1	Secret h(k(SN,DS),IDm,SN,DS,n1,n2)	Ok	Verified	No attacks.
		enc_update,SN2	Alive	Ok	Verified	No attacks.
		enc_update,SN3	Weakagree	Ok	Verified	No attacks.
		enc_update,SN4	Niagree	Ok	Verified	No attacks.
		enc_update,SN5	Nisynch	Ok	Verified	No attacks.
	DS	enc_update,DS1	SKR h(k(SN,DS),IDm,SN,DS,n1,n2)	Ok	Verified	No attacks.
		enc_update,DS2	Alive	Ok	Verified	No attacks.
		enc_update,DS3	Weakagree	Ok	Verified	No attacks.
		enc_update,DS4	Niagree	Ok	Verified	No attacks.
		enc_update,DS5	Nisynch	Ok	Verified	No attacks.

Done.

Figure 5: Scyther Verification Result for Proposed ENC Update Procedure

6 Conclusion

In our study, we analyze security vulnerabilities in the data integrity in the existing ENC update system and propose a PBFT-based security protocol to complement them. The proposed protocol maintains compatibility with the IHO standard (S-63, S-101). Additionally, protocol ensures digital signature and session key agreement and adds a Consensus layer within the ENC update procedure. Based on this procedure, we ensure data integrity, reliability, and synchronization between the communicating nodes.

The proposed protocol performed key agreement using a pre-shared key for compatible with existing standards. This method enhances portability and efficient ENC data updates through session key establishment. In addition, we checked core security properties such as mutual authentication, secure session key, and ensured integrity using a formal verification tool, Scyther. The proposed protocol can effectively defend against major threats such as forgery, tampering, rollback attacks, and key leakage, and it has been proven to provide reliable security even in real maritime environments.

The results of this paper can be extended to future S-101-based next-generation ENC standards, as well as to field tests in real-world maritime operating environments to evaluate practical feasibility and performance. Future research will also explore optimizing the consensus process by adopting alternative consensus algorithms such as Tendermint, HoneyBadger, and HotStuff. Incorporating these algorithms is expected to further enhance transaction throughput, latency efficiency, and fault tolerance, thereby improving the robustness and scalability of the proposed framework in practical deployment scenarios.

Acknowlegement

This work was supported by the Glocal University 30 Project Fund of Gyeongsang National University in 2025.

References

Castro, M., & Liskov, B. (1999). Practical Byzantine fault tolerance. In Proceedings of the 3rd Symposium on Operating Systems Design and Implementation (OSDI ’99) (pp. 173–186). USENIX Association.

Hrcak. (2024). Cyber security risks in ECDIS and mitigation measures. Pomorstvo, 38(2), 123–135.

International Chamber of Shipping, & BIMCO. (2020). Cyber security onboard ships: Guidelines for shipowners and operators (Version 4).

International Hydrographic Organization. (2020). S-57: IHO transfer standard for digital hydrographic data. IHO.

International Hydrographic Organization. (2022). S-100: Universal hydrographic data model. IHO.

International Hydrographic Organization. (2023a). ENC & ECDIS cyber security instruction. IHO.

International Hydrographic Organization. (2023b). S-63: IHO data protection scheme. IHO.

International Hydrographic Organization. (2023c). S-66: Facts about electronic charts and carriage requirements. IHO.

Li, W., Xu, H., & Zhou, X. (2020). PBFT-based secure data exchange protocol for Internet of Vehicles. IEEE Access, 8, 67235–67245.

Riviera Maritime Media. (2022). ECDIS cyber security: Emerging vulnerabilities and protection strategies.

Yin, M., Malkhi, D., Reiter, M. K., Gueta, G., & Abraham, I. (2019). HotStuff: BFT consensus with linearity and responsiveness. In Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing (pp. 347–356). ACM.

Zhang, Y., Wang, J., & Chen, X. (2021). A lightweight PBFT consensus mechanism for industrial IoT. Sensors, 21(14), 4764.

Hegde, P., Rajendran, S., & Bhat, S. (2023). Securing medical IoT networks using an enhanced PBFT consensus with EigenTrust and verifiable random functions. IEEE Access, 11, 105233–105246.

Kumar, A., Sharma, R., & Gupta, P. (2023). R-PBFT: Reputation-based practical Byzantine fault tolerance for Internet of Vehicles. IEEE Internet of Things Journal, 10(15), 13452–13464.

Liu, X., & Zhu, J. (2024). An improved practical Byzantine fault tolerance algorithm for aggregating node preferences. Scientific Reports, 14, 31200.

Wu, X., Zhang, L., & Chen, Y. (2025). DBPBFT: Dual-blockchain practical Byzantine fault tolerance for scalable IoT systems. IEEE Transactions on Industrial Informatics, 21(4), 4821–4834.

Xu, J., Wang, H., & Li, Q. (2023). ABC-GSPBFT: Artificial bee colony-based group scoring PBFT consensus for aviation data sharing. IEEE Transactions on Intelligent Transportation Systems, 24(9), 9854–9867.

Zhang, Z., Li, M., & Zhou, Y. (2024). NR-PBFT: Network resource-optimized PBFT for maritime cold-chain tracking. Sensors, 24(3), 945.

Bothur, D. (2017). A critical analysis of security vulnerabilities and countermeasures in smart ship systems. Proceedings of the Australasian Conference on Information Security and Privacy.

Kim, D., Zand, A., & Woo, M. (2018). Measuring revocation effectiveness in the Windows code-signing PKI. In USENIX Security Symposium (pp. 863–880). USENIX Association.

Oruc, A., & Aydin, M. (2025). Perspectives on the cybersecurity of the integrated navigation system (INS). Journal of Marine Science and Engineering, 13(6), 1087.

Svilicic, B., Kamahara, J., Celic, J., & Bolm, M. (2019). Raising awareness on cyber security of ECDIS. Proceedings of the TransNav Conference.

Kayişoğlu. G., Güneş, B, & Bolat, P. (2024). ECDIS cyber security dynamics analysis based on the Fuzzy-FUCOM method. Transactions on Maritime Science, 13(2), 455–470.

Zhu, L., Hu, Z., Heidemann, J., Mankin, A., Wessels, D., & Hoffman, P. (2016). Measuring the latency and pervasiveness of TLS certificate revocation. In Proceedings of the ACM Internet Measurement Conference (pp. 101–114). ACM.

Cremers, C. J. F. (2008). *The Scyther tool: verification, falsification, and analysis of security protocols*. Proceedings of the 20th International Conference on Computer Aided Verificatior (CAV'08), Princeton, New Jersey, USA. https://doi.org/10.1007/978-3-540-70545-1_38