sensors

MDPI

Article

A Security Information Transmission Method Based on DHR for Seafloor Observation Network

Fei Ying [1,2], Shengjie Zhao [1,2,3,4,*] and Jia Wang [5]

[1] College of Electronic and Information Engineering, Tongji University, Shanghai 201804, China; 23yingfei@tongji.edu.cn
[2] Key Laboratory of Embedded System and Service Computing, Ministry of Education, Shanghai 201804, China
[3] School of Software Engineering, Tongji University, Shanghai 201804, China
[4] Engineering Research Center of Key Software Technologies for Smart City Perception and Planning, Ministry of Education, Shanghai 200003, China
[5] School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China
* Correspondence: shengjiezhao@tongji.edu.cn

Abstract: A seafloor observation network (SON) consists of a large number of heterogeneous devices that monitor the deep sea and communicate with onshore data centers. Due to the long-distance information transmission and the risk of malicious attacks, ensuring the integrity of data in transit is essential. A cryptographically secure frame check sequence (FCS) has shown great advantages in protecting data integrity. However, the commonly used FCS has a collision possibility, which poses a security risk; furthermore, reducing the encryption calculation cost is a challenge. In this paper, we propose a secure, lightweight encryption scheme for transmitted data inspired by mimic defense from dynamic heterogeneous redundancy theory. Specifically, we use dynamic keys to encrypt a data block and generate multiple encrypted heterogeneous blocks for transmission. These continuously changing encrypted data blocks increase the confusion regarding the original encoded data, making it challenging for attackers to interpret and modify the data blocks. Additionally, the redundant information from the multiple blocks can identify and recover tampered data. Our proposed scheme is suitable for resource-constrained environments where lightweight encryption is crucial. Through experimental demonstrations and analysis methods, we determine the effectiveness of our encryption scheme in reducing computational costs and improving security performance to protect data integrity.

Keywords: DHR architecture; seafloor observation network; privacy security

check for updates

Citation: Ying, F.; Zhao, S.; Wang, J. A Security Information Transmission Method Based on DHR for Seafloor Observation Network. *Sensors* **2024**, *24*, 1147. https://doi.org/10.3390/s24041147

Academic Editor: Hai Dong

Received: 23 November 2023
Revised: 23 January 2024
Accepted: 23 January 2024
Published: 9 February 2024

1. Introduction

A seafloor observation network (SON) is an emerging platform for human observation of the ocean. A SON consists of various wire-connected seafloor sensors working collaboratively to monitor vast deep-sea environments (Figure 1). As a permanent infrastructure, cabled SON can provide abundant power and broad bandwidth communication [1]. It enables all-weather, in situ, continuous, real-time, and high-precision observation of the ocean from the sea floor to the sea surface, which is crucial to the development of marine science [2].

A SON requires the collection, storage, transmission, and processing of massive amounts of marine data from sensors to operation centers. To ensure efficient transmission, junction boxes serve as relay nodes, processing fragmented data from cable-connected sensors into structured data blocks and transmitting them to onshore stations. However, long-distance data transmission makes unintentional (e.g., packet loss) and intentional (e.g., tampering attacks) errors or changes to the data more likely, which can be difficult to detect [3]. Consequently, mechanisms are needed to ensure secure information transmission from the seafloor to the shore. Notably, onshore stations are designed with sufficient buffer and computing resources to store received data and handle altered or missing data.

Therefore, the security of information transmission in SONs can be considered within the integrity of each individual data block inside a packet over long distances.

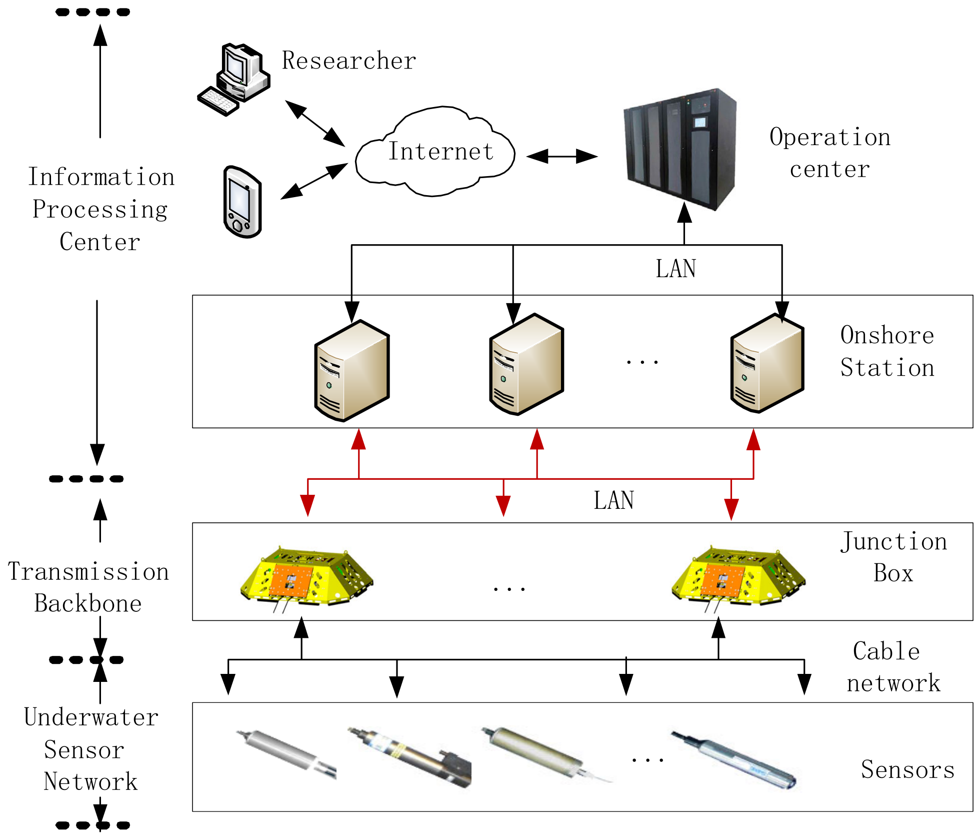


Figure 1. Information monitoring system in seafloor observatory network.

A frame check sequence (FCS), which adds redundancy or additional information to data blocks, is a common method for checking data integrity and detecting errors or changes in received data blocks. The three most frequently used techniques for generating FCS values are watermarking schemes, cyclic redundancy check (CRC), and cryptography algorithms [4]. Watermarking schemes offer lightweight data integrity schemes by inserting a secret piece of information, called a watermark, to detect changes in the original data stream [5]. These methods require redundant bits in the data to embed the watermark, which could be a weakness if the underwater sensors do not support modification of sent data blocks. Message authentication codes (MACs) [6] require the sender and receiver to share a secret key to verify the message's integrity. However, these mechanisms work well only within medium-scale networks [7], as SONs consist of numerous sensors and relay nodes, and the vast data transfer might render the key unavailable. Cryptographic CRC checksums are another common way to secure data integrity with minimal extra resources [8]. As shown in Figure 2, CRC is susceptible to collisions. Considering the large-scale data transmission required by SONs, it is impossible to avoid CRC collisions, leading to security risks due to data integrity issues. In light of the aforementioned challenges and considerations, the core issue we aim to address in this paper pertains to ensuring the integrity of transmitted data. Specifically, we focus on managing the challenges posed by large volumes and the need for timely handling of delay-sensitive data. Our objective is to safeguard the data transmitted between seafloor sensors and onshore operation centers within SONs, thereby preventing potential threats such as tampering and unauthorized data access.

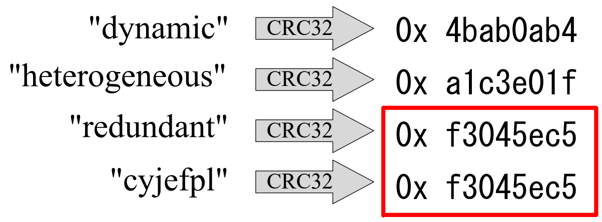


Figure 2. Collision example under CRC check (The CRC checksum values of 'redundant' and 'cyjefpl' are identical).

In this paper, we present an efficient and secure method for information transmission within a SON. Our approach concentrates on ensuring the secure transmission of data between junction boxes and onshore stations. The junction box compiles data from underwater sensors and constructs a secure data block for transmission to the onshore station. We employ a dynamic heterogeneous redundant (DHR) framework as a security measure, utilizing heterogeneity and redundancy to defend against various attack types. This framework is applicable to numerous applications, such as computer networks, distributed systems, and cybersecurity defenses [9]. Inspired by the DHR-based active defense framework, we encrypt data blocks with dynamic keys and generate multiple encrypted heterogeneous data blocks for transmission. These encrypted data blocks, as variants of the original data block, are expected to be decoded at the receiver with the same content. Inconsistency in the decoded content implies that the received data block has been altered. The redundant encrypted multi-blocks enhance security by increasing the attacker's complexity since they cannot interpret the data using a single encrypted block. The main contribution of the paper can be summarized as follows:

1. We present a novel approach to safeguarding data transmission within a SON by utilizing a DHR framework. Our method's simplicity and low computational complexity make it well-suited for deployment in SON devices with limited computational capabilities. To the best of our knowledge, this is the first instance of employing a DHR framework for this purpose;
2. We introduce an active defense framework that uses dynamic key encryption to encrypt data blocks and generates heterogeneous data blocks during transmission. This method significantly increases the difficulty for attackers trying to decipher the information, as a single encrypted block is insufficient for interpretation, thus enhancing the overall security of data integrity;
3. Experimental results provide evidence that the proposed framework effectively defends against data tampering and data-stealing attacks within a SON environment.

The paper is structured as follows: In Section 2, we introduce the structure of a SON and discuss the associated security risks. Section 3 provides a comprehensive review of the related works that serve as the foundation for our proposed method. In Section 4, we delve into the problem formulation and provide insights into the motivations behind our research. Section 5 is dedicated to the detailed explanation of our proposed method and includes an in-depth analysis of its performance. To validate the effectiveness of our approach in enhancing security defense, we present experimental results in Section 6. In Section 7, we summarize our research contributions and conclude the paper. Furthermore, we explore potential directions for future research in this field.

2. Background

2.1. Seafloor Observation Network

As shown in Figure 3, a SON consists of both surface components (i.e., onshore data centers, surface stations) and underwater components (i.e., junction boxes and sensors). It enables long-term, large-scale monitoring of deep-sea regions. Specifically, the surface components supply power to the underwater components via optical cables and analyze the collected data. For safety purposes, the control unit at an onshore data center processes the data and automatically cuts off power when the warning system detects abnormal values. At the same time, the junction box in the underwater components converts high voltage to medium voltage, providing power to the underwater sensors and transmitting the collected information to the land station [10]. Common underwater sensors include acoustic Doppler current profilers (ADCP), hydrophones, conductivity–temperature–depth (CTD) sensors, and ocean-bottom seismographs (OBS). Numerous underwater sensors connect to a junction box in linear, tree-like, or ring-like configurations, forming an underwater sensor network.

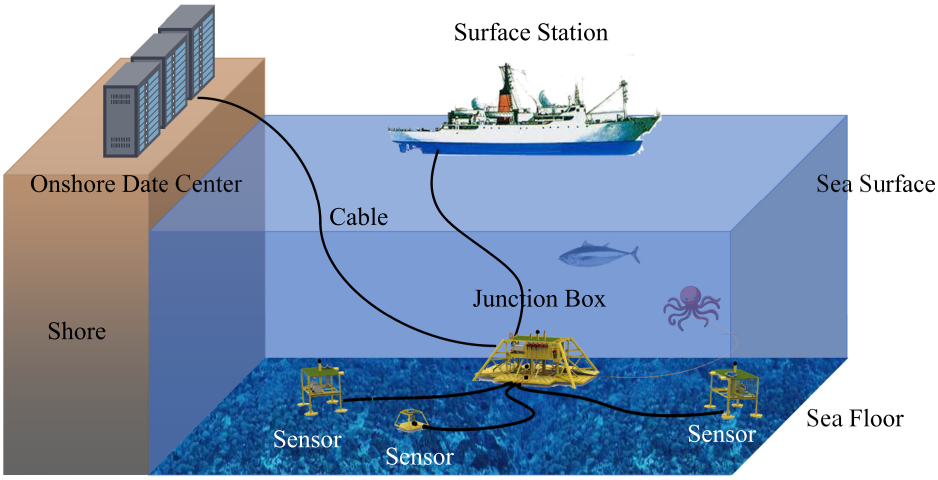


Figure 3. The illustration of a seafloor observation network.

2.2. Security Risks in SONs

It is important to note that SONs face several security risks. Cable providers may introduce backdoors or embed monitoring equipment and triggers in cable components before deployment [11]. Unauthorized or malicious use of these interfaces can lead to data leakage during transmission. Additionally, network management systems typically depend on HTTP or TCP/IP protocols for connections, which makes it easy for attackers to intercept protocol packets and analyze or obtain data information [12]. Attackers may also gain control over sensor nodes to steal or tamper with sensitive data [13]. Altered data could cause system failures, resulting in power cutoffs or fault isolation at onshore data centers, thereby disrupting continuous underwater environmental observations. Moreover, data leaks or tampering could pose serious threats to ocean observations and lead to critical decision-making errors. These significant risk concerns in SONs have hindered the advancement of seabed scientific researches.

3. Related Works

In this section, we review the methods to ensure data integrity and DHR applications that are closely related to our work.

3.1. Methods for Ensuring Data Integrity

Data integrity refers to the accuracy, validity, and consistency of information within a system. When transmitting data, especially over an unstable media (e.g., deep sea environment), several potential security issues arise, such as physical failure and malicious tampering. Ensuring data integrity is crucial to prevent data contamination, fraudulent data injection, and data manipulation [14]. Several technologies, including error-detecting codes, cryptography algorithms, arbitration schemes, and watermarking schemes, are frequently employed to address data integrity issues [15].

Error-detecting codes are widely used techniques in both wired and wireless networks, ensuring that only correctly marked frames are forwarded to higher-level communication protocols, while frames with errors are discarded. CRC, checksums, and MAC are a few examples of error-detecting codes. Among these, CRC stands out as particularly effective, employing binary division instead of addition. Standardized polynomials, such as CRC-16 and CRC-32, are common variants of CRC; however, when selecting a specific CRC polynomial, it is crucial to consider the trade-off between security and computational cost [8]. In the field of data integrity, several methods use CRC to ensure the reliability and accuracy of data. For instance, Chen et al. [16] reduced the overall cost of the prevention and repair stage in distributed systems by implementing redundant error correction codes and network coding. Yu et al. [17] ensured data integrity by using identity CRC, providing an effective way to protect privacy data. Similarly, Ateniese et al. [18] employed a forward error-correcting code to enhance the performance of data processing frameworks. Despite their advantages, encoding-based methods can lead to increased computational costs and reduced running efficiency when using large security keys and blocks.

Encrypting data during transmission can protect its integrity. Various encryption methods exist, including symmetric encryption, asymmetric encryption, and hash mapping [19]. Symmetric encryption employs a single key for both encryption and decryption, while asymmetric encryption uses a pair of public and private keys for enhanced security. Hash mapping transforms data into a fixed-sized hash through mathematical methods. Common encryption methods include the Caesar cipher [20], Data Encryption Standard (DES) [21], Triple Data Encryption Standard (3DES) [22], Advanced Encryption Standard (AES) [23], and BlowFish [24]. However, when data is transmitted through encryption, the security of the encrypted data becomes vulnerable if the key is lost or stolen.

Arbitration is another method for protecting the accuracy and completeness of data through third-party verification. Data arbitration can be categorized into the following two main security models: provable data possession (PDP) and proof of retrievability (PoR). PDP includes static and dynamic schemes. The static PDP scheme focuses on protecting the security of confidential data, but it lacks the capability to restore lost data [25]. Meanwhile, the dynamic PDP scheme focuses on dynamic data updates, enabling recovery of some lost data by incorporating error-correcting codes [26]. However, the arbitration process faces challenges concerning privacy data breaches and authentication of third-party identities.

Watermarking-based techniques aim to provide lightweight solutions for data integrity and authentication, which embeds a secret piece of information, known as a watermark, into the original data streams to detect any alterations. In recent years, they have been widely used in data transmission to prevent private information from being illegally obtained [27]. Al-Shayea et al. [28] proposed a new watermarking method based on the use of orthogonal families to withstand various types of attacks. Ferdowsi et al. [29] applied deep learning technology to dynamic watermarking to identify attack threats in the Internet of Things. However, attackers can easily decipher watermarking methods, and the cost of computation remains high. Furthermore, several watermarking techniques require the addition of extra bits in the data stream to embed the watermark, posing a vulnerability if the transmission does not support the data distortions.

3.2. DHR Architecture and Applications

DHR architecture is an endogenous security technique, as depicted in Figure 4. Within this framework, the input agent plays a crucial role in distributing input requests to a diverse set of heterogeneous redundant executors, each responsible for independent processing. Subsequently, the processing results undergo a multimodal voting process, and only the consistently matching results are chosen as the final output. This approach significantly reduces the risk of security weaknesses and vulnerabilities being exploited, thereby ensuring the trustworthiness of the system results. This architecture is widely adopted in the domain of endogenous security. Wei et al. [30] proposed a mimic web application security technology based on the DHR architecture, which makes it difficult for attackers to maintain continuous control and access after a successful attack. Yu et al. [31] successfully applied the DHR architecture to industrial network security, effectively increasing the difficulty of exploiting backdoors, such as paralysis, rule tampering, and information theft.

Furthermore, DHR architecture's adaptability is evident from its successful implementation in various domains, including the Internet of Vehicles [32] and edge networks [33]. These real-world deployments have demonstrated the versatility and effectiveness of DHR architecture in guarding against potential security threats.

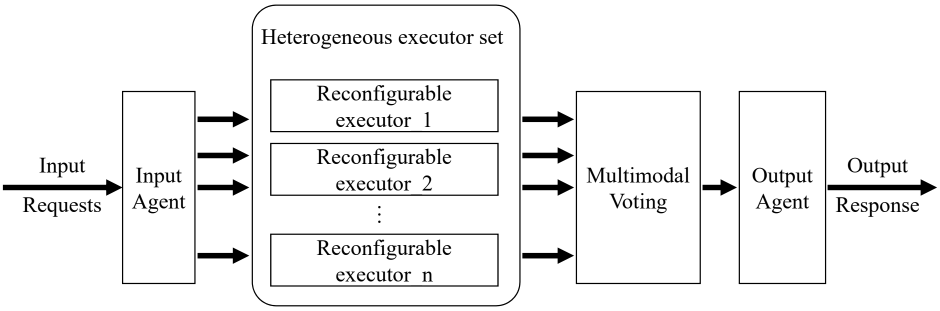

Input
Requests
Input
Agent
Heterogeneous executor set
Reconfigurable executor 1
Reconfigurable executor 2
⋮
Reconfigurable executor n
Multimodal
Voting
Output
Agent
Output
Response

Figure 4. The overview of DHR architecture.

4. Preliminaries

4.1. Notations and Problem Formulation

Let $\mathcal{E} = \{E_1(), E_2(), \ldots, E_n()\}$ be a set of n mapping functions. The i-th sender encrypts a message I using a mapping function $E_i() \in \mathcal{E}$, resulting in ciphertext $E_i(I)$ transmitted to the receiver. We use A_{send} to denote the information space and $A_{recv}^{E_i}$ is the encrypted space based on $E_i()$. The receiver decrypts the ciphertext using the inverse function $E_i^{-1}()$. The above process satisfies the following properties:

- Invertibility: For any $I \in A_{send}$, there exists a unique message $I' \in A_{recv}$ such that $E_i(I) = I'$ and $E_i^{-1}(I') = I$;
- Redundancy: For any $I \in A_{send}$, where $E_i() \neq E_j()$, there exists the encrypted information $E_i(I) = E_j(I)$;
- Uniqueness: For any $I' \in A_{recv}$ such that $E_i^{-1}() \neq E_j^{-1}()$, then decoded information $E_i^{-1}(I') \neq E_j^{-1}(I')$.

Our objective is to protect data integration by encrypting information using the above mapping functions. For clarity, we summarize the frequently used notations in Table 1:

Table 1. Frequently used notations and descriptions.

Notations	Descriptions
$E_i()$	The i-th encryption function
$E_i^{-1}()$	The i-th decryption function
A_{send}	The set of all the plain text
A_{recv}	The set of all the ciphertext
I	Plain text to be sent
I'	Ciphertext received

4.2. Security Assumptions

The security of the proposed scheme is based on the following two attack problems:

- Data tampering attack: Refers to unauthorized changes made to data blocks while they are being transmitted. This attack is considered successful if the attacker is able to modify the data without detection by the system;
- Data stealing attack: Occurs when attackers gain access to a network and steal sensitive data while it is in transit.

4.3. Motivation

The limited number of mapping functions in $\mathcal{E}$ poses a security risk for SONs as they are rarely updated in reality. Attackers can exploit unknown vulnerabilities to launch brute-force attacks and guess the mapping function used for data transmission. To prevent such attacks, we propose equipping random perturbation parameters λ to the encryption result of mapping functions, denoted as $E_i(\cdot;\lambda)$, where λ is the key randomly selected from a pool Λ.

5. Methodology

In this section, we introduce the system model and its application in SONs for ensuring information transmission integrity.

5.1. System Model

The DHR architecture is a security approach that leverages heterogeneity and redundancy to protect systems against various types of attacks. As shown in Figure 5, the system model of a DHR-based security framework involves the following three main entities: distribute module, heterogeneous encryption module, and decryption module.

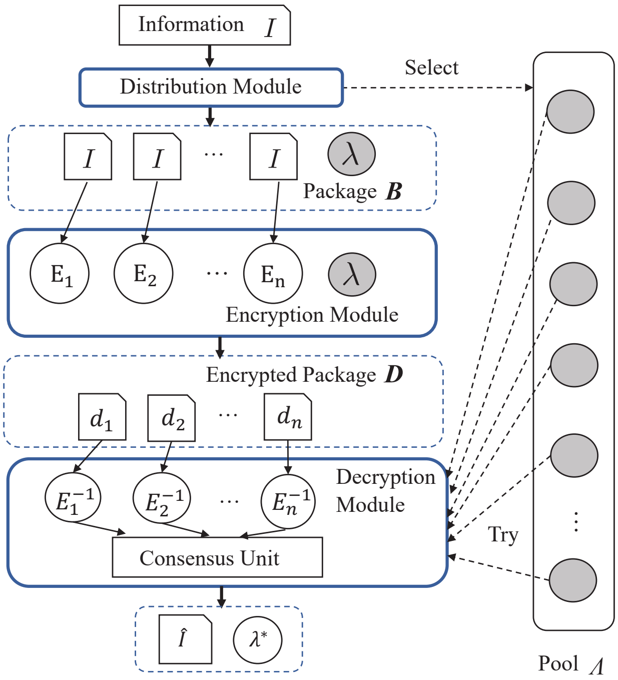


Figure 5. The illustration of system model.

Distribution module takes an input I and generates a package $\mathcal{B} = [\mathcal{I}, \lambda]$ by duplicating I into a set $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$ of n identical copies, and selecting a random key λ from a pool Λ. The resulting package $\mathcal{B}$ is then forwarded to the encryption unit for further processing.

Encryption module comprises n encryption units, each utilizing a unique mapping function $E_i() \in \mathcal{E}$ and the received key λ to encrypt the i-th element of $\mathcal{I}$. The resulting encrypted ciphertext package is denoted as $\mathcal{D}$, where $d_i \in \mathcal{D}$ represents the encryption of the i-th element of $\mathcal{I}$ using the i-th encryption unit and the received key, i.e., $d_i = E_i(I_i, \lambda)$.

Decryption module consists of n decryption units and a consensus unit. Each decryption unit attempts to decrypt the ciphertext package $\mathcal{D}$ using the corresponding decryption mapping function by trying each key in a pool Λ. Specifically, for a given key λ, the output of the i-th decryption unit is denoted as $\hat{I}_i = E_i^{-1}(d_i, \lambda)$, where $E_i^{-1}()$ is the inverse function of the encryption function used to encrypt the data d_i. The consensus unit then compares the outputs from all decryption units and selects the key λ^* that yields consistent outputs across all units. This is performed by maximizing the consensus function Γ over all keys in the pool: $\lambda^* = \text{argmax}_{\lambda \in \Lambda} \Gamma(E_1^{-1}(d_1, \lambda), E_2^{-1}(d_2, \lambda), \ldots, E_n^{-1}(d_n, \lambda))$, where $\Gamma(\cdot)$ is a consensus function that evaluates the congruence of outputs generated by each decryption unit, given a specific key λ. The outcome of $\Gamma(\cdot)$ is a quantified score that reflects the degree of consensus among the decryption outputs. Consequently, the selected key λ^* is responsible for maximizing this consensus score. The final decrypted output is attained by applying the identified optimal key λ^* to one of the decryption units that produces the consensual output.

5.2. Application in SONs

In this section, we detail the implementation of our proposed security method for transmitting information in SONs. As illustrated in Figure 6, our system architecture is composed of the sending end, located on the junction boxes, and the receiving end, positioned at the onshore station. The sending end comprises a distribution module (DiM) and an encryption module (EnM) composed of three heterogeneous shift units. These components play a crucial role in ensuring the security of the transmitted information. On the other hand, the receiving end consists of three reversion units functioning as the decryption module (DeM) and a consensus unit. Their primary task is to recover the transmitted information by decrypting the received data. To provide a deeper understanding of our system architecture and its components, we present the details of each module in the subsequent subsections.

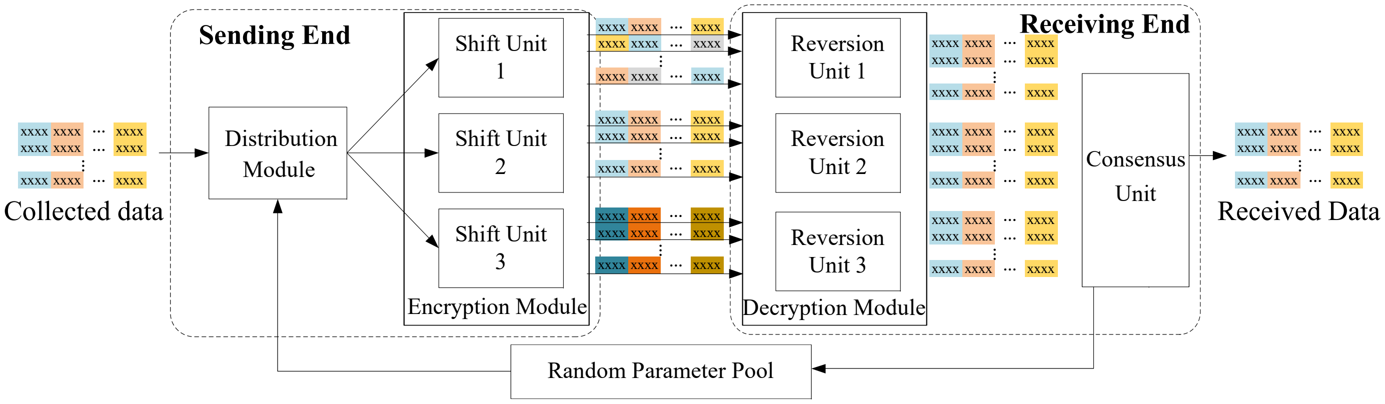


Figure 6. The systematic implementation of a seafloor observation network. (The data from the same sensor are shown in the same color).

5.2.1. Distribution Module

SONs organize sensor data into messages consisting of n blocks of 4-bit hexadecimal numbers. In our framework, these messages are initially transmitted to a DiM, which receives n blocks at a time. The DiM then applies the stacking blocks method [34] to combine messages from m sensors and reorganize them into the stacked packets denoted by $\mathcal{D}$. As illustrated in Figure 7, data blocks from the same sensor are listed in the same column. To enhance security, the stacked packets are concurrently dispatched to multiple shift units within EnM, along with the temporary dynamic key λ selected from a key pool Λ.

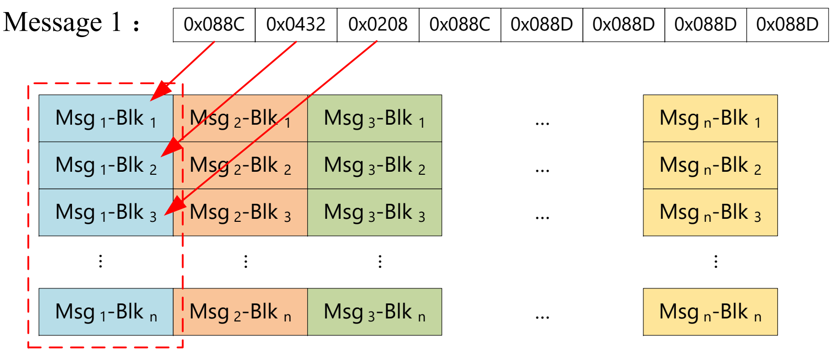


Figure 7. Message structure and stacked packets. (The data from the same sensor are shown in the same color).

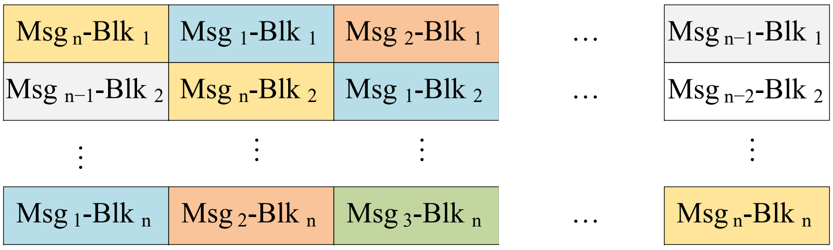
5.2.2. Encryption Module

To minimize energy consumption during package encryption in the junction box and ensure cost-effectiveness for practical DHR-based applications [35], we employ a three-degree redundancy approach. Our encryption module comprises three heterogeneous shift units that use distinct shift strategies to transform data packet $\mathcal{D}$ into corresponding $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{D}_3$. Specifically, the three shift strategies are as follows:

- Horizontal strategy. For the data packet $\mathcal{D}$, we shuffle the columns of the data blocks using encryption parameters λ while keeping the rows of each data block. This forms an encrypted data packet $\mathcal{D}_1^{'}$, where data block $d_{ij} \in \mathcal{D}$ will be translated to the position $d_{i',j'}$, where i' and j' satisfy the following conditions:

$$\begin{cases} i' = i \\ j' = (i + j + \lambda - a) \bmod n. \end{cases} \tag{1}$$

where a a is a natural number constant. When λ sets a, the data packet after horizontal translation is shown in Figure 8;



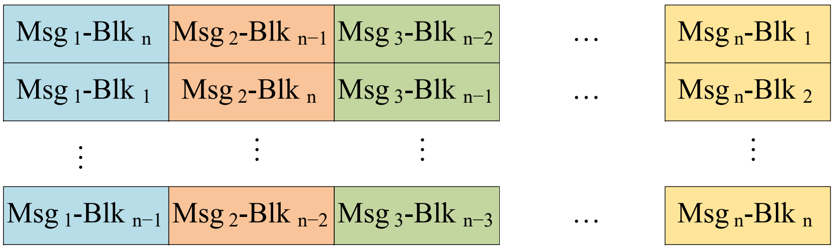
Msg $_{n}$-Blk $_{1}$	Msg $_{1}$-Blk $_{1}$	Msg $_{2}$-Blk $_{1}$	…	Msg $_{n-1}$-Blk $_{1}$
Msg $_{n-1}$-Blk $_{2}$	Msg $_{n}$-Blk $_{2}$	Msg $_{1}$-Blk $_{2}$	…	Msg $_{n-2}$-Blk $_{2}$
⋮	⋮	⋮		⋮
Msg $_{1}$-Blk $_{n}$	Msg $_{2}$-Blk $_{n}$	Msg $_{3}$-Blk $_{n}$	…	Msg $_{n}$-Blk $_{n}$

Figure 8. The data package encryption with the horizontal strategy ($\lambda = a$).

- Vertically translation strategy. The second shift strategy involves vertically translating the data blocks, where the columns of each block are preserved while the rows are shuffled using encryption parameters λ to produce the encrypted data packet $\mathcal{D}'$. For a given data block d_{ij} in the stacked data packet D, it will be shifted to position $d_{i',j'}$, where i' and j' are determined based on following equation:

$$\begin{cases} i' = (i + j + \lambda - a) \bmod n \\ j' = j, \end{cases} \tag{2}$$

where i and j represent the row and column of a data block in the original data packet, while i' and j' correspond to the row and column of the data block in the translated data packet. When λ equals a, the resulting vertically translated data packet is illustrated in Figure 9;



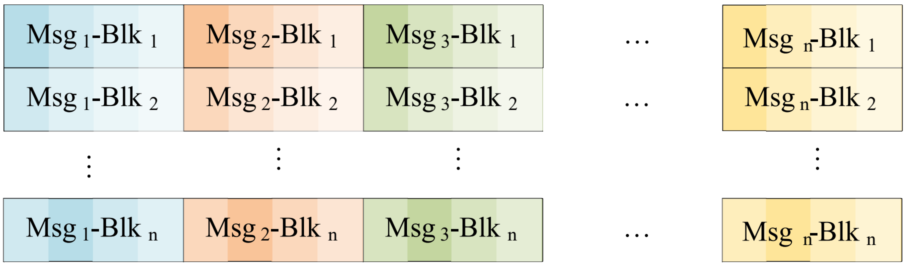
Msg $_{1}$-Blk $_{n}$	Msg $_{2}$-Blk $_{n-1}$	Msg $_{3}$-Blk $_{n-2}$	…	Msg $_{n}$-Blk $_{1}$
Msg $_{1}$-Blk $_{1}$	Msg $_{2}$-Blk $_{n}$	Msg $_{3}$-Blk $_{n-1}$	…	Msg $_{n}$-Blk $_{2}$
⋮	⋮	⋮		⋮
Msg $_{1}$-Blk $_{n-1}$	Msg $_{2}$-Blk $_{n-2}$	Msg $_{3}$-Blk $_{n-3}$	…	Msg $_{n}$-Blk $_{n}$

Figure 9. The data package encryption with the vertically strategy ($\lambda = a$).

- Numerical strategy. This strategy involves using a parameter λ to add a translation offset to the binary representation of the numerical value of each data block in data packet $\mathcal{D}$. Since the information collected by the seabed observation sensor is comprised of 4-bit hexadecimal numbers, the data block d_{ij} in $\mathcal{D}$ is transformed into $d_{i',j'}$ using the following formula:

$$d'ij = (((dij)_{10} + i + j + \lambda - a) \bmod 16^4)_{16}, \tag{3}$$

where $(\cdot)_{10}$ denotes decimal conversion, and $(\cdot)_{16}$ denotes hexadecimal conversion. The encrypted packet employing the numerical strategy is illustrated in Figure 10.



Msg $_1$-Blk $_1$	Msg $_2$-Blk $_1$	Msg $_3$-Blk $_1$	…	Msg $_n$-Blk $_1$
Msg $_1$-Blk $_2$	Msg $_2$-Blk $_2$	Msg $_3$-Blk $_2$	…	Msg $_n$-Blk $_2$
⋮	⋮	⋮		⋮
Msg $_1$-Blk $_n$	Msg $_2$-Blk $_n$	Msg $_3$-Blk $_n$	…	Msg $_n$-Blk $_n$

Figure 10. The data package encryption with the numerical strategy.

After undergoing processing by the heterogeneous encryption unit, the original data packet $\mathcal{D}$ is partitioned into the following three distinct packets: $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{D}_3$, which are then transmitted through the cable to the onshore data center or surface station. The integration of redundant shift operations effectively hinders attackers from deducing the original data values even if they intercept the transmission and possess prior knowledge of the collection process. This significantly raises the difficulty level for attackers attempting to steal authentic sensing data.

5.2.3. Decryption Module

The DeM is located at the onshore station and is responsible for decrypting the received encrypted package. It tries different shifting parameters λ' from the pool Λ and uses the corresponding reverse rules to shift each data block in the package. By comparing the consistency of the three restored data blocks, the parameter λ^* used at the encryption module can be determined, and the transmitted packet can be decrypted. The decryption process is shown in Algorithm 1.

Algorithm 1 The decryption process.

```
function DECRYPTION($\mathcal{D}_1$,$\mathcal{D}_2$,$\mathcal{D}_3$, Pool $\Lambda$)
    for $\lambda \in \Lambda$ do
        $\mathcal{D}'_1 \leftarrow$ horizonBack( $\mathcal{D}_1, \lambda$); /*Reversed horizontal translation strategy;/*
        $\mathcal{D}'_2 \leftarrow$ verticalBack( $\mathcal{D}_2, \lambda$); /*Reversed vertical translation strategy;/*
        $\mathcal{D}'_3 \leftarrow$ numBack( $\mathcal{D}_3, \lambda$); /*Reversed numerical translation strategy;/*
        c $\leftarrow$ consistency($\mathcal{D}'_1, \mathcal{D}'_2, \mathcal{D}'_3$); /*Score the level of agreement;/*
        if $\max_c < c$ then
            $\max_c \leftarrow c, \lambda^* \leftarrow \lambda$ /*Get $\lambda^*$ that maximizes $c$;*/
    $\mathcal{D} \leftarrow$ horizonBack( $\mathcal{D}_1, \lambda^*$) or $\mathcal{D} \leftarrow$ verticalBack( $\mathcal{D}_2, \lambda^*$) or $\mathcal{D} \leftarrow$ numBack( $\mathcal{D}_3, \lambda^*$)
    return $\mathcal{D}$
```

If a single sensor data is transmitted with an error or under a tampering attack, the encryption feedback controller is triggered, prompting the *distribution module* to select a new random parameter λ' from the pool for *encryption module* to shift the data packet for the upon the arrival data package, causing the translation rules of each data packet to change. This renders the previously observed data pattern unusable for the attacker, preventing them from continuing the attack experience. In general, the randomness of the new key λ' selected from the parameter pool Λ avoids an attacker launching a tampering attack. Thus, we use Equation (4) to quantify the randomness of the parameter key selection.

$$H(\Lambda) = -\sum_{\lambda_i \in \Lambda} p(\lambda_i) log(p(\lambda_i)), \tag{4}$$

where $p(\lambda_i)$ to represent the probability of obtaining the key λ_i. When each key in the emulation parameter pool has the same probability of selection, $H(\Lambda)$ reaches its maximum value, indicating that the randomness of the emulation parameter selection is highest and the defense effect of the emulation-based data security system is best. The feedback controller can also defend against replay attacks.

5.3. Security Analysis

This subsection analyzes the behavior of our framework in the presence of an attack. Figure 6 illustrates a sequence of encrypted information packages transmitted through SONs. Throughout this subsection, we use the following notation: $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ represents the encrypted stacked packets through *Shift Units* through horizontal, vertical, and numerical strategies, respectively. $d^c_{i,j}$ refers to the i-th data block belonging to the j-th sensor in the c-th stacked packet, where $c \in \{1, 2, 3\}$.

To decrypt encrypted packets, an attacker needs to know the random parameters used to encrypt data packets at the current moment. In other words, an attacker can decrypt encrypted packets at any time by choosing one parameter. Its success probability is constant, and it does not rise as the attacker's data collection increases. In contrast, when using techniques like hash functions, the probability of success for the attackers rises as they gather more data.

5.3.1. Tampering Packet

If the $d^1_{i,j}$ block is tampered with, the modifications affect some bytes in $d^1_{i,j}$. In this case, an integrity error is detected through the consensus unit at the receiving end. This error is detected because the output of the three reversion unit at $d_{i,j}$ is not the same. Therefore, this data block can be recovered if the output from the two remaining reversion unit outputs are identical at this position. Otherwise, retransmissions are requested for resending the $d^1_{i,j}$ values where there are consensus errors. If the maximum number of attempts is reached, the $d^1_{i,j}$ block is discarded.

5.3.2. Data Theft

In this scenario, attackers listen to the data information transmitted in the LAN and analyze the collected information to obtain real data. As the values at the data block $d_{i,j}$ are changed over time, for example, at time t_a, the value at the position of the i-th data block of the j-th sensor is $d^1_{i,j}$, and at time t_b, the value at that position is $d^1_{i+x,j+y}$. This prevents attackers from obtaining the genuine sensor data by relying solely on the absolute packet positions. On the other hand, at different times, the distance between the real data reflected by $d^1_{i,j}$ and $d^1_{i+1,j}$ is different, which hinders attackers from extracting sensitive information using relative packet positions.

6. Implementation and Evaluation

This section outlines the hardware and software utilized in the implementation of the proposed schemes. Then, the experimental findings are discussed.

6.1. Environmental Setup

We conduct our experiments in the environment as shown in Figure 11. The hardware test-bed includes a METS sensor, a CTD sensor, and a DO sensor connected to a Raspberry Pi (i.e., junction box) that encrypts the sensing data and transmits it via cable to a ThinkSystem (i.e., onshore sever). The information is subsequently decoded by the onshore server. To attack the system, the attacker uses the router to obtain access to the network. Table 2 provides the details of the environment and configuration used in the experiment. The environment sensors are connected to the Raspberry Pi (i.e., junction box) through the RS422 serial port. The Raspberry Pi (i.e., junction box) is linked with the ThinkSystem (i.e., onshore sever) through the RJ-45 interface to form an Ethernet LAN. Additionally, we simulate an attacker who can access the network through a router and carry out attacks. Typically, the data-stealing attack is capable of capturing data packets transmitted in the LAN, while the data tampering attack could modify data packets transmitted in the LAN to deceive the receiver.

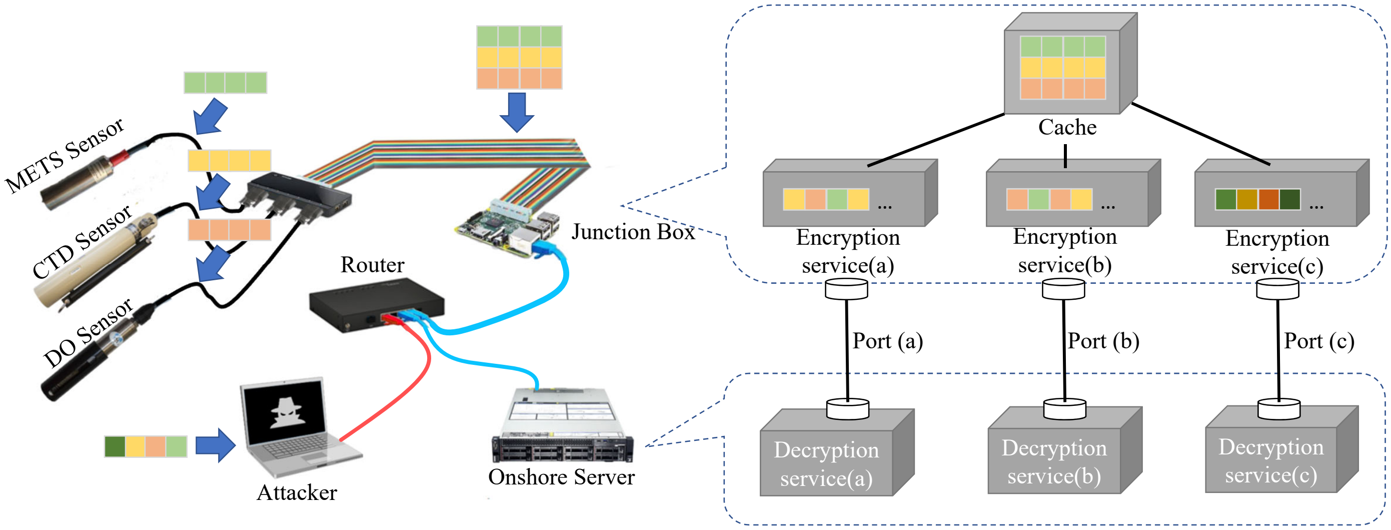


Figure 11. The experimental environment.

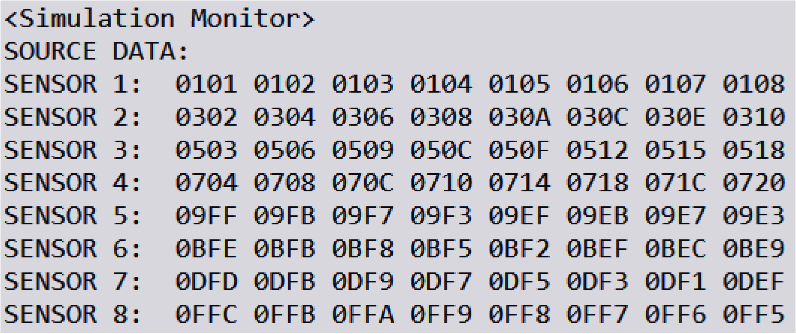
Table 2. The hardware's detailed information.

Name	Model	Function	Configuration
METS Sensor	Franatech Classic METS [Reppenstedt, Germany]	Methane inspection	Measurement range of 50 nMol/L to 10 µMol/L
CTD Sensor	SAIV AS SD204 [Bergen, Norway]	Record seawater conductivity, salinity, temperature, depth, and sound speed (water density)	Salinity range: 0–40 ppt, Temperature range: −2–40 °C, Depth range: 500–6000 m
DO Sensor	Edaphic ES-O2-DW [Moorabbin, Australia]	Measure the oxygen in gas	Oxygen range: 0–20 mg/L
Raspberry Pi	Raspberry Pi 3 Model B [Shenzhen, China]	Simulate a junction box for encrypting collected sensor data	CPU: 64-bit quad-core, ARM Cortex-A53, Memory: 1 GB
Server	ThinkSystem SR558H [Beijing, China]	Simulate the operations center and onshore station to encrypt the information	CPU: Hygon C86 5280, Memory: 32 GB
Router	LS1008G V2 [Shenzhen, China]	Provide basic network topology	8 Ports, 10/100/1000 Mbps

6.2. *Simulated Man-in-the-middle*

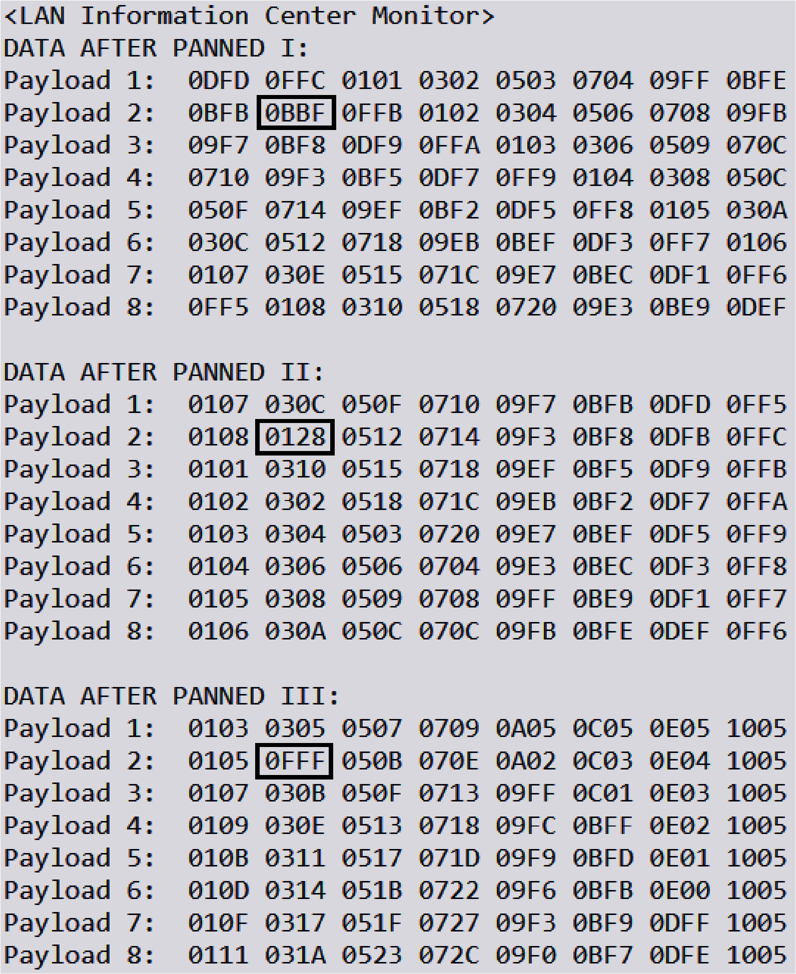
We simulate a man-in-the-middle (MITM) attack to evaluate the efficacy of our proposed security scheme in detecting unauthorized modifications to data. In this scenario, an attacker intercepts and randomly alters certain data blocks before they reach their intended destination. The security scheme should be able to identify these modifications and correct them. Figure 12 provides an example of a data packet transmitted from a Raspberry Pi to the server. The packet comprises eight blocks, each containing a 4-bit hexadecimal number that represents data collected from eight individual sensors. Upon initiating the simulated MITM attack, the altered packets in the secured overlay network are depicted in Figure 13.

The tampered data blocks are highlighted with boxes to indicate unauthorized changes made by the attacker.



```
<Simulation Monitor>
SOURCE DATA:
SENSOR 1:  0101 0102 0103 0104 0105 0106 0107 0108
SENSOR 2:  0302 0304 0306 0308 030A 030C 030E 0310
SENSOR 3:  0503 0506 0509 050C 050F 0512 0515 0518
SENSOR 4:  0704 0708 070C 0710 0714 0718 071C 0720
SENSOR 5:  09FF 09FB 09F7 09F3 09EF 09EB 09E7 09E3
SENSOR 6:  0BFE 0BFB 0BF8 0BF5 0BF2 0BEF 0BEC 0BE9
SENSOR 7:  0DFD 0DFB 0DF9 0DF7 0DF5 0DF3 0DF1 0DEF
SENSOR 8:  0FFC 0FFB 0FFA 0FF9 0FF8 0FF7 0FF6 0FF5
```

Figure 12. Data collected by simulated sensors.



```
<LAN Information Center Monitor>
DATA AFTER PANNED I:
Payload 1:  0DFD 0FFC 0101 0302 0503 0704 09FF 0BFE
Payload 2:  0BFB 0BBF 0FFB 0102 0304 0506 0708 09FB
Payload 3:  09F7 0BF8 0DF9 0FFA 0103 0306 0509 070C
Payload 4:  0710 09F3 0BF5 0DF7 0FF9 0104 0308 050C
Payload 5:  050F 0714 09EF 0BF2 0DF5 0FF8 0105 030A
Payload 6:  030C 0512 0718 09EB 0BEF 0DF3 0FF7 0106
Payload 7:  0107 030E 0515 071C 09E7 0BEC 0DF1 0FF6
Payload 8:  0FF5 0108 0310 0518 0720 09E3 0BE9 0DEF

DATA AFTER PANNED II:
Payload 1:  0107 030C 050F 0710 09F7 0BFB 0DFD 0FF5
Payload 2:  0108 0128 0512 0714 09F3 0BF8 0DFB 0FFC
Payload 3:  0101 0310 0515 0718 09EF 0BF5 0DF9 0FFB
Payload 4:  0102 0302 0518 071C 09EB 0BF2 0DF7 0FFA
Payload 5:  0103 0304 0503 0720 09E7 0BEF 0DF5 0FF9
Payload 6:  0104 0306 0506 0704 09E3 0BEC 0DF3 0FF8
Payload 7:  0105 0308 0509 0708 09FF 0BE9 0DF1 0FF7
Payload 8:  0106 030A 050C 070C 09FB 0BFE 0DEF 0FF6

DATA AFTER PANNED III:
Payload 1:  0103 0305 0507 0709 0A05 0C05 0E05 1005
Payload 2:  0105 0FFF 050B 070E 0A02 0C03 0E04 1005
Payload 3:  0107 030B 050F 0713 09FF 0C01 0E03 1005
Payload 4:  0109 030E 0513 0718 09FC 0BFF 0E02 1005
Payload 5:  010B 0311 0517 071D 09F9 0BFD 0E01 1005
Payload 6:  010D 0314 051B 0722 09F6 0BFB 0E00 1005
Payload 7:  010F 0317 051F 0727 09F3 0BF9 0DFF 1005
Payload 8:  0111 031A 0523 072C 09F0 0BF7 0DFE 1005
```

Figure 13. Heterogeneous data transmitted in LAN.

6.3. Security Analysis Metrics

We utilize the previously mentioned experimental setup to simulate data packet transmission in a SON. Specifically, we continuously transmit 10,000 data packets and assess the experimental results based on the following three key metrics:

Receive Accuracy: To evaluate the effectiveness of our proposed defense mechanism against data tampering attacks, we deliberately alter varying numbers of data blocks within each packet. The receive accuracy metric quantifies the proportion of data blocks received correctly. A higher receive accuracy implies a stronger defense against tampering.

Similarity: To evaluate the system's ability to resist data stealing attacks, we compare randomly captured data packets during transmission with their original packets. We calculate similarity using the average longest common subsequence (LCS) and the Hamming distance. Lower similarity values indicate stronger defense capabilities against data-stealing attacks.

Numerical Offset: To evaluate the effectiveness of preventing attackers from concealing encryption patterns from potentially intercepted packets, we determine the numeric difference between data blocks in the original and encrypted packets. A larger numerical offset indicates a lower likelihood of attackers discerning encryption patterns through analysis of intercepted data.

6.4. Evaluation of Anti-Tampering Ability

To evaluate the anti-tampering capability of our proposed DHR-based security system against data packet tampering attacks, we conduct experiments in which we randomly tamper with some data blocks in the data packet. We then analyze whether the sensor processing service of a SON received tampered data under different transmission methods. The results of these experiments are presented in Figure 14. Our method achieves 99.02% receive accuracy when 2% of data blocks are tampered with. The receive accuracy decreases when more than 10% of data blocks are tampered with. The experimental findings demonstrate that the anti-tampering ability of the CRC check method declines significantly as the data tampering rate increases. The shuffling overlapped method (SOM) enhances the anti-tampering ability to some extent, while our proposed method delivers the best performance. This is attributed to the combination of heterogeneity, redundancy, and dynamic adaptation within the DHR framework, rendering it highly resilient against a diverse array of attacks. Even if an attacker manages to compromise one or more data blocks, the remaining blocks can continue to provide protection, maintaining the overall security of the system.

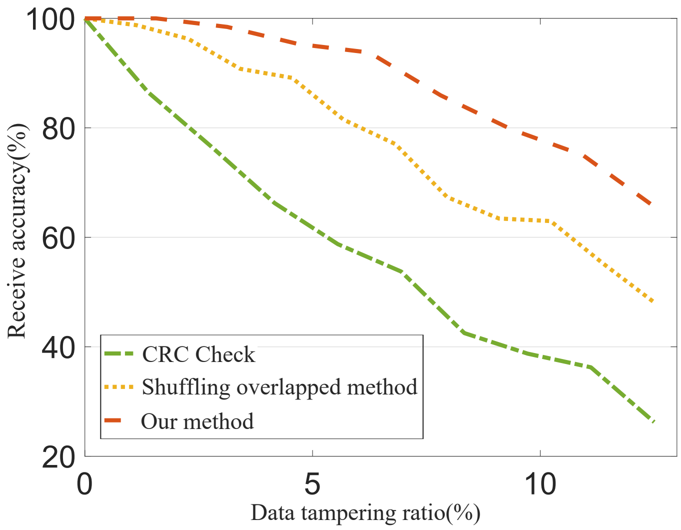


Figure 14. Evaluation of anti-tampering capability.

6.5. Evaluation of Anti-Stealing Capability

To analyze system's efficacy against data stealing, we evaluate the average discrepancy between stolen data and its real value. We use Hamming distance and longest common subsequence (LCS) distance as measurement metrics. A greater Hamming distance signifies a larger dissimilarity between stolen and origin data, while a smaller LCS distance implies that attackers can obtain less information through stealing [36]. The experimental results displayed in Figure 15 reveal that with the CRC-16 check method, the LCS distance between the encrypted and original data is 16, while the minimum Hamming distance is 0. In contrast, our data encryption security scheme yields an average LCS distance of 1.3

and an average minimum Hamming distance of 15.3. These results surpass those of the SOM method, indicating that the ciphertext generated by our encryption module exhibits sufficient heterogeneity.

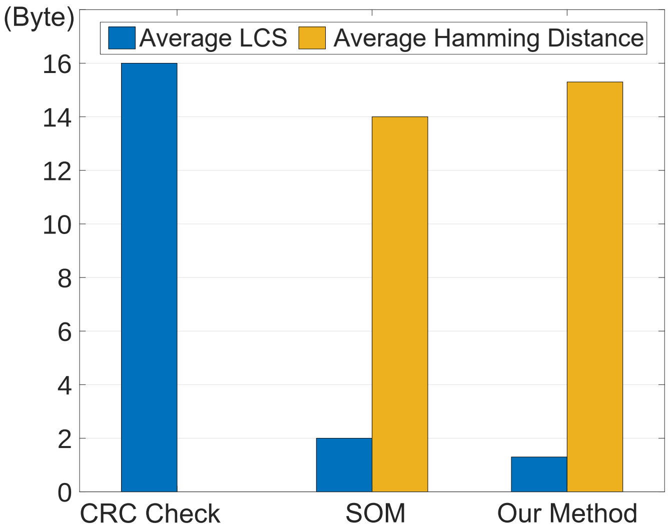


Figure 15. Data theft resistance under different transmission modes.

Furthermore, we analyze the likelihood of attackers identifying encryption patterns from intercepted data. We achieve this by encrypting 10,000 packets from 3 sensors (i.e., METS sensor, CTD sensor, and DO sensor) and computing the block-shifting offset. The resulting distribution is depicted in Figure 16. It can be observed that the offset is uniformly distributed, without any recognizable pattern (i.e., Block1 has the same chance to be shifted to other blocks). This suggests that it is challenging for attackers to detect patterns and infer the original observation data. The findings demonstrate the efficacy of our security system in preventing data stealing. Consequently, it is considerably difficult for attackers to decipher the encryption rules of data packets through extended observation.

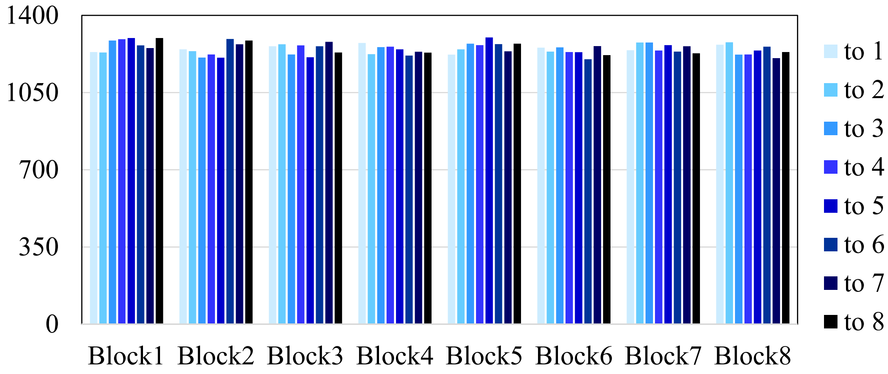


Figure 16. Frequency of heterogeneous data block offset.

6.6. Evaluation on Side-Channel Attacks

In light of potential side-channel attacks exploiting information gathered from a system's physical characteristics, such as power consumption, our method stabilizes the amplitude of encrypted data. To validate this, we assessed the attacker's ability to extract information. Specifically, we used the Pair-HMM method [37], to cluster 1000 pieces of encrypted data from each of the 8 sensors based on numerical values and compared it with the clustering of the original data. Figure 17 shows that when clustering the encrypted data produced by our method, it does not directly reveal the true value range of the original sensor data. The correlation coefficients between the stolen data and the original data resulted in value of [−0.2523, −0.4334, −0.5611, 0.1329, −0.3443, −0.9993, and −0.3444]. This confirms that our approach effectively increases the difficulty for attackers attempting physical attacks, such as side-channel attacks.

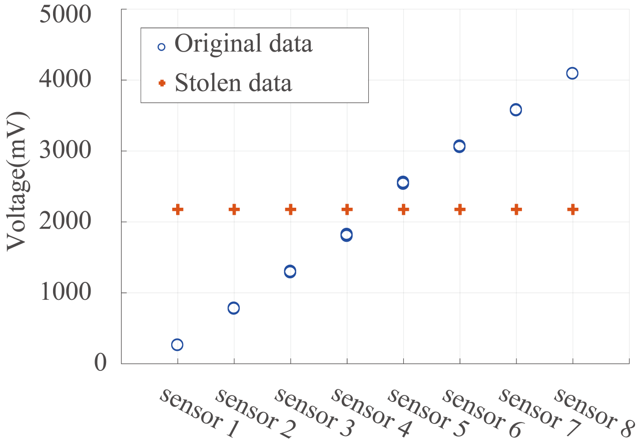


Figure 17. Comparison of theft data clustering centers and the observed data.

6.7. System Overhead Analysis

We select 8 sensors, each transmitting 16 bytes of observation data, to analyze data parity overhead during transmission. Typically, we employ our encryption method along with error-detecting codes (i.e., CRC-16, CRC-32, and SOM), hash mapping (i.e., MD5), and symmetric encryption (i.e., DES and AES) methods to encrypt the data on the Raspberry Pi, ensuring that all measurements are taken under the same configurations. Then, we conduct experiments to compare the resulting overhead length and execution time. Table 3 presents the execution time and overhead length results. We implement checksums and shifting operations using the NesC language on the TinyOS operating system. Processing times are measured using TinyOS's LocalTimeMicroC components. In addition, our results demonstrate that our method outperforms CRC-16, CRC-32, and SOM. These methods require 52.8%, 92.2%, and 303.6% more execution time compared to our method, respectively. In terms of encryption algorithms, MD5, DES, and AES take 549.3×, 990.1×, and 20,520.5× more than our method, respectively. This highlights the efficiency of our method in lightweight and effective packet transmission without extra checksums while our DHR method includes redundant data transmission, which results in bandwidth consumption, it is crucial to notice that SONs are characterized by their wired connections and sufficient bandwidth resources. Consequently, our approach remains the preferred choice for enhancing information transmission security within SONs.

Table 3. Overhead data comparison (bytes) and execution time (µs).

	Data	**Overload**	**Overload (%)**	**Time**
CRC-16 [6]	128	16	12.5	5.12 (+52.8%)
CRC-32 [38]	128	32	25	6.44 (+92.2%)
SOM [34]	128	48	37.5	13.52 (+303.6%)
MD5 [19]	128	128	100	1840.11 (×549.3)
DES [21]	128	64	50	3317.08 (×990.1)
AES [23]	128	128	100	68,743.82 (×20,520.5)
Our Method	128	256	200	3.35

7. Conclusions

In this paper, we propose an innovative security information transmission approach for SONs based on dynamic heterogeneous redundancy theory. SONs consist of diverse devices that monitor the deep sea and communicate with onshore data centers. Due to extended data transmission distances and the network's susceptibility to malicious attacks, ensuring data integrity is important. Our method employs dynamic keys to encrypt data blocks, generating multiple encrypted heterogeneous blocks for transmission. These dynamically changing encrypted blocks enhance confusion and diffusion of the original data by utilizing shuffle and shift operations. This significantly complicates attackers' attempts to interpret or manipulate the data. Moreover, the redundancy within the multiple blocks assists in the identification and recovery of tampered data. Through empirical demonstrations in a minimal system, we validate the effectiveness of our approach in reducing data transmission errors and enhancing data integrity during transmission. In the future, we plan to apply our method to larger, more complex SONs to thoroughly evaluate their scalability and performance. Furthermore, we will certainly consider exploring the integration of our method with other security frameworks in our future work. For instance, we plan to leverage fuzzy neural networks to capture complex patterns and relationships within the data, aiming to enhance encryption quality and effectively manage high-dimensional data transmission.

Author Contributions: Conceptualization, F.Y. and S.Z.; methodology, F.Y. and S.Z.; software, F.Y.; validation, F.Y.; formal analysis, F.Y.; investigation, F.Y.; resources, F.Y. and S.Z.; writing—original draft preparation, F.Y. and J.W.; writing—review and editing, F.Y. and J.W.; supervision, S.Z.; funding acquisition, S.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Research and Development Project under Grant 2019YFB2102300, in part by the National Natural Science Foundation of China under Grant 61936014, in part by Shanghai Municipal Science and Technology Major Project No. 2021SHZDZX0100, in part by the Fundamental Research Funds for the Central Universities, in part by the China National Scientific Seafloor Observatory.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Acknowledgments: We would sincerely want to thank the peoples in the China National Scientific Seafloor Observatory who are supported to do this work.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Fenghua, L.; Yanguo, L.; Haibin, W.; Yonggang, G.; Fei, Z. Research progress and development trend of seafloor observation network. *Bull. Chin. Acad. Sci.* **2019**, *34*, 321–330. (In Chinese)
2. Yu, Y.; Xu, H.; Xu, C. An object model for seafloor observatory sensor control in the east China sea. *J. Mar. Sci. Eng.* **2020**, *8*, 716. [CrossRef]
3. Xie, H.; Yan, Z.; Yao, Z.; Atiquzzaman, M. Data collection for security measurement in wireless sensor networks: A survey. *IEEE Internet Things J.* **2018**, *6*, 2205–2224. [CrossRef]
4. Xiao, H.; Zheng, B.; Isshiki, T.; Kunieda, H. Hybrid shared-memory and message-passing multiprocessor system-on-chip for UWB MAC layer. *IET Comput. Digit. Tech.* **2017**, *11*, 8–15. [CrossRef]
5. Wazirali, R.; Ahmad, R.; Al-Amayreh, A.; Al-Madi, M.; Khalifeh, A. Secure watermarking schemes and their approaches in the IoT technology: An overview. *Electronics* **2021**, *10*, 1744. [CrossRef]
6. Noh, J.; Jeon, S.; Cho, S. Distributed blockchain-based message authentication scheme for connected vehicles. *Electronics* **2020**, *9*, 74. [CrossRef]
7. Bello, L.L.; Steiner, W. A perspective on ieee time-sensitive networking for industrial communication and automation systems. *Proc. IEEE* **2019**, *107*, 1094–1120. [CrossRef]
8. Tsimbalo, E.; Fafoutis, X.; Piechocki, R.J. CRC error correction in IoT applications. *IEEE Trans. Ind. Inform.* **2016**, *13*, 361–369. [CrossRef]
9. Wu, J. Cyberspace endogenous safety and security. *Engineering* **2022**, *15*, 179–185. [CrossRef]

10. Pulvirenti, S.; Schmelling, J.-W.; D'Amico, A.; Giorgio, E.; Aurnia, S. Idmar infrastructure: The junction box and shore station optical network. In Proceedings of the 2022 IEEE International Workshop on Metrology for the Sea, Learning to Measure Sea Health Parameters (MetroSea), Milazzo, Italy, 3–5 October 2022; pp. 61–65.
11. Hummelholm, A. Undersea optical cable network and cyber threats. In Proceedings of the European Conference on Information Warfare and Security, Academic Conferences International, Coimbra, Portugal, 4–5 July 2019; pp. 650–659.
12. Eleftherakis, D.; Vicen-Bueno, R. Sensors to increase the security of underwater communication cables: A review of underwater monitoring sensors. *Sensors* **2020**, *20*, 737. [CrossRef]
13. Bueger, C.; Liebetrau, T. Protecting hidden infrastructure: The security politics of the global submarine data cable network. *Contemp. Policy* **2021**, *42*, 391–413. [CrossRef]
14. Rattan, A.K. Data integrity: History, issues, and remediation of issues. *PDA J. Pharm. Sci. Technol.* **2018**, *72*, 105–116. [CrossRef]
15. Tan, C.B.; Hijazi, M.H.A.; Lim, Y.; Gani, A. A survey on proof of retrievability for cloud data integrity and availability: Cloud storage state-of-the-art, issues, solutions and future trends. *J. Netw. Comput. Appl.* **2018**, *110*, 75–86. [CrossRef]
16. Chen, B.; Curtmola, R.; Ateniese, G.; Burns, R. Remote data checking for network coding-based distributed storage systems. In Proceedings of the 2010 ACM Workshop on Cloud Computing Security Workshop, Chicago, IL, USA, 8 October 2010; pp. 31–42.
17. Yu, Y.; Au, M.H.; Ateniese, G.; Huang, X.; Susilo, W.; Dai, Y.; Min, G. Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage. *IEEE Trans. Inf. Forensics Secur.* **2016**, *12*, 767–778. [CrossRef]
18. Ateniese, G.; Burns, R.; Curtmola, R.; Herring, J.; Khan, O.; Kissner, L.; Peterson, Z.; Song, D. Remote data checking using provable data possession. *ACM Trans. Inf. Syst. Secur.* **2011**, *14*, 1–34. [CrossRef]
19. Gheorghiu, V.; Mosca, M. Benchmarking the quantum cryptanalysis of symmetric, public-key and hash-based cryptographic schemes. *arXiv* **2019**, arXiv:1902.02332.
20. Gowda, S.N. Innovative enhancement of the Caesar cipher algorithm for cryptography. In Proceedings of the 2016 2nd International Conference on Advances in Computing, Communication, & Automation (ICACCA) (Fall), Bareilly, India, 30 September–1 October 2016; pp. 327–330.
21. Alenezi, M.N.; Alabdulrazzaq, H.; Mohammad, N.Q. Symmetric encryption algorithms: Review and evaluation study. *Int. J. Commun. Netw. Inf. Secur.* **2020**, *12*, 256–272.
22. Adhie, R.P.; Hutama, Y.; Ahmar, A.S.; Setiawan, M. Implementation cryptography data encryption standard (DES) and triple data encryption standard (3DES) method in communication system based near field communication (NFC). *J. Phys. Conf. Ser.* **2018**, *954*, 012009.
23. Abdullah, A.M. Advanced encryption standard (AES) algorithm to encrypt and decrypt data. *Cryptogr. Netw. Secur.* **2017**, *16*, 11.
24. Alabaichi, A.; Ahmad, F.; Mahmod, R. Security analysis of blowfish algorithm. In Proceedings of the 2013 Second International Conference on Informatics & Applications (ICIA), Lodz, Poland, 23–25 September 2013; pp. 12–18.
25. Ateniese, G.; Burns, R.; Curtmola, R.; Herring, J.; Kissner, L.; Peterson, Z.; Song, D. Provable data possession at untrusted stores. In Proceedings of the 14th ACM Conference on Computer and Communications Security, Alexandria, VA, USA, 31 October–2 November 2007; pp. 598–609.
26. Yuan, J.; Yu, S. Public integrity auditing for dynamic data sharing with multiuser modification. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 1717–1726. [CrossRef]
27. Araghi, T.K.; Megías, D.; Rosales, A. Evaluation and analysis of reversible watermarking techniques in wsn for secure, lightweight design of iot applications: A survey. In Proceedings of the Advances in Information and Communication: 2023 Future of Information and Communication Conference (FICC), San Francisco, CA, USA, 2–3 March 2023; Volume 2, pp. 695–708.
28. Al-Shayea, T.K.; Mavromoustakis, C.X.; Batalla, J.M.; Mastorakis, G.; Mukherjee, M.; Chatzimisios, P. Efficiency-aware watermarking using different wavelet families for the Internet of Things. In Proceedings of the ICC 2019–2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–6.
29. Ferdowsi, A.; Saad, W. Deep learning-based dynamic watermarking for secure signal authentication in the internet of things. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6.
30. Wei, D.; Xiao, L.; Shi, L.; Yu, L. Mimic web application security technology based on dhr architecture. In Proceedings of the International Conference on Artificial Intelligence and Intelligent Information Processing (AIIIP 2022), Qingdao, China, 17–29 June 2022; Volume 12456, pp. 118–124.
31. Yu, F.; Wei, Q.; Geng, Y.; Wang, Y. Research on key technology of industrial network boundary protection based on endogenous security. In Proceedings of the 2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), Chongqing, China, 18–20 June 2021; Volume 4, pp. 112–121.
32. Tang, M. Research on edge network security technology based on DHR. In Proceedings of the 2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA), Dalian, China, 20–21 August 2022; pp. 614–617.
33. Chen, L.; Miao, Y.; Yu, C.; Liu, S. CD-DAA-MD: A cross-domain DAA scheme with Mimic Defense for Internet of Vehicles. In Proceedings of the 2022 IEEE 28th International Conference on Parallel and Distributed Systems (ICPADS), Nanjing, China, 10–12 January 2023; pp. 139–146.
34. Velasco, F.; Palomares, J.; Olivares, J. Lightweight method of shuffling overlapped data-blocks for data integrity and security in WSNs. *Comput. Netw.* **2021**, *199*, 108470. [CrossRef]
35. Park, S.-H.; Kim, J.-Y.; Cho, I.-J.; Hwang, B.-M. Redundancy management design for triplex flight control system. *J. Korean Soc. Aeronaut. Space Sci.* **2010**, *38*, 167–179.

36. Navarro, J.; Deruyver, A.; Parrend, P. A systematic survey on multi-step attack detection. *Comput. Secur.* **2018**, *76*, 214–249. [CrossRef]
37. Wu, Z.; Shu, M.; Shi, J.; Cao, Z.; Xu, F.; Li, Z.; Xiong, G.; Yiu, S. How to reverse engineer ICS protocols using pair-HMM. In *Information and Communication Technology for Intelligent Systems: Proceedings of ICTIS 2018*; Springer: Singapore, 2019; Volume 2, pp. 115–125.
38. Abdulnabi, M.S.; Ahmed, H. Design of efficient cyclic redundancy check-32 using FPGA. In Proceedings of the 2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE), Khartoum, Sudan, 12–14 August 2018; pp. 1–5.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.