applied sciences

MDPI

Article

REISCH: Incorporating Lightweight and Reliable Algorithms into Healthcare Applications of WSNs

Mishall Al-Zubaidie [1,2,*], Zhongwei Zhang [2] and Ji Zhang [2]

[1] Department of Computer Science, Education College for Pure Science, Thi-Qar University, Nasiriyah 64001, Iraq

[2] Faculty of Health, Engineering and Sciences, University of Southern Queensland, Toowoomba, QLD 4350, Australia; Zhongwei.Zhang@usq.edu.au (Z.Z.); Ji.Zhang@usq.edu.au (J.Z.)

* Correspondence: u1070801@umail.usq.edu.au or Mishall.Al-Zubaidie@usq.edu.au; Tel.: +61-469-869-029

Received: 3 February 2020; Accepted: 11 March 2020; Published: 15 March 2020

Abstract: Healthcare institutions require advanced technology to collect patients' data accurately and continuously. The tradition technologies still suffer from two problems: performance and security efficiency. The existing research has serious drawbacks when using public-key mechanisms such as digital signature algorithms. In this paper, we propose **R**eliable and **E**fficient **I**ntegrity **S**cheme for Data **C**ollection in **H**WSN (REISCH) to alleviate these problems by using secure and lightweight signature algorithms. The results of the performance analysis indicate that our scheme provides high efficiency in data integration between sensors and server (saves more than 24% of alive sensors compared to traditional algorithms). Additionally, we use Automated Validation of Internet Security Protocols and Applications (AVISPA) to validate the security procedures in our scheme. Security analysis results confirm that REISCH is safe against some well-known attacks.

Keywords: ECDSA; healthcare; homomorphic; integrity; pseudonym; WSN

1. Introduction

Medical records of patients require accurate, secure, and efficient electronic systems to be managed and organized. Electronic medical record (EMR) systems are extremely useful for managing patients' data. These systems are widely disseminated in the health sector [1]. Moreover, EMR systems need patients' data collection technology such as a wireless sensor network (WSN). This technology consists of a group of sensing nodes that communicate wirelessly with each other to gather data about a particular environment in various applications. A WSN often has limited resources such as energy and memory, but it provides comfort, speed, accuracy and safety to humans by monitoring a specific area without human intervention or presence [2]. An important application that has brought the attention of sensor networks to many researchers is healthcare (HC), because of great importance in our lives to reduce the effects of diseases on the health of patients. Providing better HC quality of lower cost will be a key aim of all health industries over the next decades [3].

These applications relying on the use of WSNs are known as healthcare wireless sensor networks (HWSNs) [4]. By using HWSNs, healthcare providers including physicians, doctors, and nurses can access data and information about patients on an ongoing basis, whether at clinics or in hospitals. Therefore, this medical record leads to more accurate diagnosis and thus is likely to lead to an improvement in the patient's condition. For many diseases that require constant monitoring and precise care, HWSN is the best method used by doctors to get patients' data, as this technology provides patients with comfort and more care at less cost [2]. Since these applications monitor patients' activities without interruption, accurately and continually, they should lead to an improvement in their health condition [5].

Disclosure of medical records for patients in the HC systems results in weak security in these systems. In addition, some security mechanisms such as public key signing significantly affect WSN performance. Therefore, several issues need to be addressed when designing schemes for collecting data in HC applications. These issues are critical to the acceptance and success of HC applications in the healthcare sector. These issues listed are as follows:

- **Communications security:** To protect data and information between source and destination, security mechanisms, such as signatures should be applied to prevent an attacker from accessing records transferred between network entities (for example, sensor, Cluster Head (CH), and Base Station (BS)/Local Server (LS)). These mechanisms should resist attacks such as disclosure, alteration, replication, collision, preimage, and impersonation of medical records transmitted. The communication channel should be protected end-to-end at both the wireless level and on the Internet through the integration of a set of security mechanisms and privacy [6,7].
- **Datasets security:** Medical records stored on the EMR server as a repository become the target of malicious attacks. In particular, if a HC application is based on a single server, the process of hacking this server results in both data and information being detected [8]. Besides, access to datasets without pseudonyms and signature mechanisms makes it easy for attackers to detect users' real identities (IDs). To protect users' medical records, the EMR server should not contain real information for users to prevent detection of users' identities or tampering with datasets. For instance, an EMR server contains only signatures and pseudonyms and users' identifiers are stored on the remote server, such as an Attributes Server (AS). Furthermore, the database should be available to legitimate users at any time and from anywhere, and should support authorization requests for access to partial data from an EMR repository and patients' history from a remote server, such as a Data Server (DS) [9].
- **Performance of collection devices:** WSN requires efficient security algorithms to work efficiently. EMR systems use WSN to collect patient data. However, a WSN is source-constrained in terms of energy, computing and memory. Therefore, when using signature mechanisms, security and performance should be efficient. The efficiency of these algorithms is a major challenge in HC applications. Namely, the sensor nodes should be very efficient to collect patient data accurately and for a long time while protecting the data collected from the penetration [10,11].

Many attacks undermine the security of the WSN of collecting patients' data and threaten the privacy of the EMR repository. These attacks have been classified into passive, active, internal, and external attacks [12–14]. For instance, potential attacks on data transferred or stored in an EMR repository by WSN are a serious risk to HC systems. As shown in Figure 1, Intruder 1 can listen to data as they are transferred from the patients' sensors or CHs to the server (LS/BS). When the attacker intercepts the message, he/she can obtain information about the physical location of the patient, ID, timestamps, source address, target address, and the medical report sent by the sensors or directed by medical staff devices. The patients' data transmitted through the sensor networks requires complete security, especially when movement through the network does not require the consent of the patient, such as moving the data of an emergency case [15]. Additionally, Intruder 2 can perform an attack on the LS/remote Central Server (CS) to penetrate the datasets to obtain patient information. An attacker can also get information from the datasets, such as the patient's name, age, address, type of disease, and the seriousness of the disease. This information allows the attacker to harm the patient in different ways, such as changing or destroying data [16]. However, designing a patients' data collection scheme with strong and heavy signing mechanisms without regard to network performance in data collection is useless and infeasible for HWSN systems. Therefore, performance and security issues are essential to provide care services in HC applications.

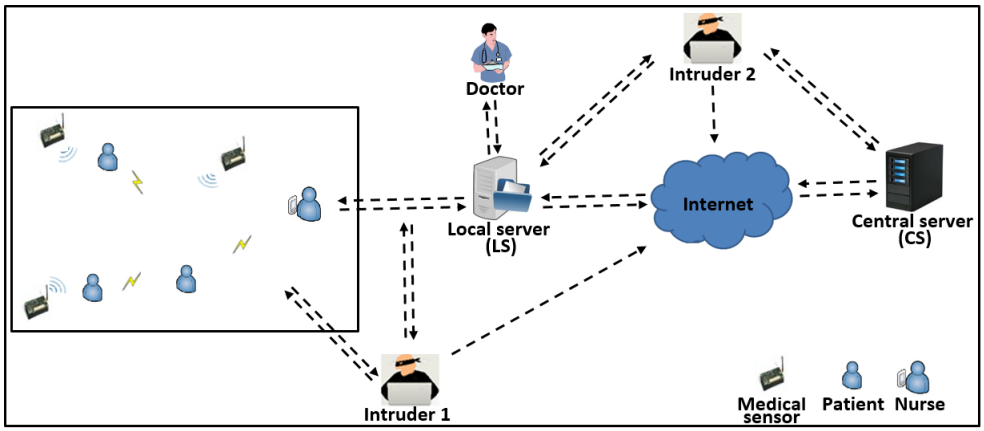

Figure 1. An attack on communication security (Intruder 1) and datasets security (Intruder 2).

1.1. Our Contributions

We propose **R**eliable and **E**fficient **I**ntegrity **S**cheme for Data **C**ollection in **H**WSN (REISCH) to ensure that patient data is transferred/stored to the *LS*/*BS* securely and efficiently. The REISCH is characterized as follows:

- REISCH applies the Elliptic Curve Digital Signature Algorithm (ECDSA) with BLAKE2bp instead of ECDSA with Secure Hash Algorithm 1 (SHA1) to improve HWSN lifetime and prevent intruders from altering/changing patients' data.
- REISCH used the homomorphic mechanism with *CH*s to reduce energy consumption when aggregating patient data from sensors.
- REISCH hides the sensor's identification (SID) and location (SL) by using random pseudonyms. This mechanism prevents intruders from detecting sensors information transmitted between network terminals.
- Formal security analysis in REISCH is simulated by an automated validation of Internet security protocols and applications (AVISPA). This tool is dramatically accepted as an effective way to validate threat models in HWSN. AVISPA is used to check that our scheme is secure against both passive and active attacks.

1.2. Paper Structure

The rest of this research is organized as follows. Section 2 discusses existing research related to our study. The trust model, threat model, and an overview of techniques used in REISCH are explained in Section 3. Section 4 provides details about the proposed data collection scheme. Section 5 discusses security and performance analysis of the REISCH scheme. Finally, Section 6 presents the conclusion and future work directions.

2. Related Existing Research

This section briefly discusses existing studies [17–25] designed to secure patient data in the EMR and highlights their drawbacks.

Fan and Gong [17] implemented ECDSA on Micaz motes with the binary field (163-bit). They improved signature verification via cooperation of the adjacent nodes. ECDSA's implementation was also presented in the sensor node (IRIS) [18]. However, because this node supported 8-bit of the microcontroller, the author modified the SHA1 code from the 32 bits original to 8 bits. Through implementation, the original algorithm is better in size and time than the modified algorithm. The author explained the possibility of using the ECDSA algorithm with the sensor node (IRIS) held 8-bit microcontroller.

To store patient data accurately, data collection schemes should rely on reliable and fast hash algorithms in ECDSA. The authors of [19] applied the ECDSA algorithm as a lightweight authentication scheme in the WSN, demonstrating the effectiveness and efficiency of using ECDSA in WSN in terms of

security and performance. Staudemeyer et al. [20] designed an ECC/ECDSA-based scheme to provide privacy in WSN. However, they did not provide a performance analysis of the security algorithms during exchange of data in WSN. Malathy et al. [21] focused on the efficiency of transmission in WSN to extend the lifetime of sensor nodes with the use of ECDSA and generated message digest (MD) with data. Their scheme relied on a colony optimization scheme to save energy in the WSN. However, it did not support privacy parameters during data transfer. Sharavanan et al. [22] proposed a scheme to monitor the heterogeneous network environments in WSN and protect the medical information of patients using ECDSA. Unfortunately, their scheme addressed only the computation processes of transport. It did not address the complicated computation processes that generate and verify the signature in ECDSA.

Recently, Sui and de Meer [23] designed a data aggregation scheme that focused on computation in demand-response management to improve performance and security efficiency. Their scheme was based on the identity signature (Bilinear Map) to protect information and data aggregated by integration and authentication. Hathaliya et al. [24] proposed an elliptic curve cryptography (160 bits) scheme to encrypt and authenticate patients' biometric properties. They used wearable sensors to collect patient data and used a mobile device to send and store these data in the medical repository (cloud server). Finally, Furtak et al. [25] designed a framework based on RSA-2048 bits and trusted modules to secure the sensors' domain and prevent unauthorized threats. They organized sensors into master, replica, and gateway categories in the network area and data structure in the sensor memory. In their framework, security procedures for the domain and sensor were used to support both integrity and authentication. Moreover, many researchers [26–30] have pointed out that ECDSA is particularly appropriate for authentication and authorization schemes because it performs lightweight processes during security procedures. Many recent studies [31–35] have also pointed out that SHA1 suffers from collision, preimage, and second preimage attacks. However, no schemes addressed SHA1 performance and security (collision, preimage, and second preimage) problems in ECDSA.

3. Preliminary Techniques for Our Data Collection Scheme

Data collection technology should be efficient and secure to meet the requirements of health institutions. The HWSN requires techniques to perform the data collection procedures before storing patients' data on the EMR server. To guarantee that only legitimate sensors are associated with the trusted *LS*, our scheme uses a set of security techniques to integrate and authenticate the collected data and detect false data in *LS*. In our scheme, we depend on algorithms that provide efficient lightweight operations and a high-security level for signature operations. This section presents the trust model, the threat model and the basic review of REISCH's techniques.

3.1. Trust Model

We assume that the history of patient data and information is stored on remote and safe servers (*CS*, *AS*, and *DS*). *LS* does not contain patients' real information. It contains only signatures and pseudonyms. Additionally, *LS* is not associated with *CH* outside a certain range. Moreover, the authorization provider of the network cannot know the correlation between patient information and data, times, and locations. Legitimate users can access partial data at *LS* without disclosing confidential patient information.

3.2. Threat Model

Building a threat model in HWSN is important to identify serious attacks on patients' data and subsequent disclosure. HWSN provides important services to the health sector compared to traditional computer networks, such as LAN and MAN, but they are more vulnerable than the latter. These networks rely on self-organization and synchronization to increase the flexible communications of sensor nodes, but HWSN suffers from a security vulnerability. Because of the wireless radio signals in WSN, it is easier for attackers to intercept data transmitted among sensors nodes, *CH*s, and *LS*. These networks are targeted for many attacks that exploit resource-constrained, untrustworthy communication and unattended processes. HWSN threats are as follows:

- The attacker performs a man in the middle (MITM) attack to modify or replay attack to resend the data to the CH/LS. The attacker's aim is to use his/her device as a legitimate sensor in the network.
- The attacker can execute a denial of service (DoS) attack on the CH/LS. This attack exploits a heavy transmission of duplicate or counterfeit data to destroy the HWSN.
- The attacker can apply several types of localization attacks such as Sybil, Wormhole, and Sinkhole to intercept of network communications.
- The attacker performs an attack to penetrate the EMR repository in the LS, to access the patient's data and reveal their identities.
- The attacker can launch an eavesdropping attack to obtain patients' data, and then perform an analysis of these data to detect the linkability among data, information, and pseudonyms.
- The attacker can copy a legitimate sensor ID in more than one counterfeit sensor. These counterfeit nodes send modified data to the network (node replication attack).
- Collision, preimage, and second preimage attacks can be implemented to change signatures and data transferred between a network's devices.

3.3. Overview of Techniques Used in REISCH

- **Electronic medical record (EMR)**

  A medical record is a communication entity used to record and review patients' health status for members of the medical staff and patients themselves. Medical records are divided into two categories: paper and electronic records. The paper record is a traditional method used to check and record patient information. This type of medical record suffered from many problems when dealing with patient data. These problems include delay, errors, lack of coordination of care equality at different levels, management of health information and data, integration of scientific evidence into HC services and decision-making practices [36], and security issues. The electronic medical record rapidly processes and transmits data across digital devices. EMRs provide HC services continuously and accurately. It has attracted the attention of both the HC industry and researchers because it provides advantages in efficiency and effectiveness. Consequently, recent studies [36–38] have indicated that the electronic medical record (EMR) reduces adverse effects among patients and providers because of its many advantages:

  1. It is easy for the patient to review his/her information/data; users can review the medical record at the same time, auto-update, and quickly retrieve information.
  2. Patient understanding of care services is improved; it facilitates patient participation and cooperation in decision-making.
  3. It reduces errors in documents and reduces the embarrassment of the patient with a professional.
  4. It increases transparent cooperation and improves the interaction between the patient and the providers.
  5. The use and quality of health information, quality of care, efficiency, cost of care, facilitate data collection, retrieval, and use of patients' data are improved.

  EMR is efficient because it provides many features and supports the use of WSN. EMR is defined as a one-organization system [39]. Currently, most Australian professionals use an EMR, which is rated similarly in several countries such as Germany, New Zealand, and the Netherlands [40]. EMR stores patient health data within a single institution and uses WSN to store patient data in a local repository for use in reports, disease diagnosis, and treatment. However, an EMR only contains a partial patient medical history [40]. For example, doctors may use an EMR to identify a patient's prescription and avoid errors, and the nurse may use an EMR to monitor tests and reports for a patient. However, if the doctor needs complete data about a patient's medical history, he/she needs to send a request to the CS. However, performance and security efficiency are the main issues when using WSN with an EMR.

- **Security properties of EMRs**

  Security of EMRs relies on the elliptic curve discrete logarithm problem (ECDLP) [41]. ECDSA utilizes small parameters which improve the performance of computations, thus diminishing process time and storage. These features are essential for large institutions and limited-resource devices such as WSN because these networks require intensive/complex processes, memory, or ower consumption [42].

  Since the intruder can tamper with the collected data (d) when they are transferred from sensors to LS, patients' data integrity is important in the HWSN environments [43]. Many reputable organizations such as NIST and IEEE use ECDSA as standard [44]. ECDSA with a 160-bit key achieves the equivalent for symmetric cryptography with a 80-bit key [45]. It is suitable for limited-resource devices because it produces small keys and provides computation speed in the integrity process. Furthermore, ECDSA uses four-point multiplication (PM) operations: one PM each for public key and signature generation and two for verification. Besides, it comprises three procedures: key generation, signature, and verification. These procedures are described as follows:

  - **Key generation:**

    1 Select a pseudorandom integer private key (K_{pr}) and compute public key (K_{pu}) $= K_{pr}G$

  - **Signature generation:**

    1 Select a pseudorandom integer k, $1 \leq k \leq n$-1.
    2 Compute e = SHA1 (d) and $kG = (x_1, y_1)$.
    3 Compute $r = x_1 \bmod n$. If $r = 0$ then go to Step 1.
    4 Compute $k^{-1} \bmod n$ and $s = k^{-1}(e + K_{pr}r) \bmod n$. If $s = 0$ then go to Step 1, else signature for the message m is (r, s).

  - **Signature verification:**

    1 Verify that r and s are integers in the interval [1,$n-1$].
    2 Compute e = SHA1 (d).
    3 Compute $w = s^{-1} \bmod n$, $u_1 = ew \bmod n$, $u_2 = rw \bmod n$ and $X = u_1G + u_2K_{pu}$.
    4 If $X = \theta$, then reject the signature. Otherwise, convert the x-coordinate x_1 of X to an integer $\bar{x}_1$, and compute $v = \bar{x}_1 \bmod n$, accept the signature if and only if $v = r$.

  ECDSA becomes inappropriate to sign d if applied poorly and incorrectly. It becomes reliable if the parameters are validated effectively [46]. In REISCH, we use ECDSA-256 bit to add a high-security level and take care to consume system resources.

- **Integrity and authentication of EMRs**

  In this subsection, we explain the two one-way hash functions, both of which are related to our study.

  - **SHA family**

    Secure Hash Algorithm (SHA) is one of the traditional hash algorithms that provides integrity and authentication when used with digital signatures. For instance, SHA1 was used in the ECDSA algorithm to perform the signature process. SHA consists of several varieties: SHA0, SHA1, SHA2, and SHA3. Both SHA2 and SHA3 consist of SHA224, SHA256, SHA384, and SHA512, but SHA3 uses a different structure than the rest of the SHA family. SHA0, SHA1 and SHA2 are built on the basis of the Merkle–Damgard structure, as shown in Figure 2 [47], and were designed by the National Security Agency (NSA). SHA3 is also known as KECCAK and is built on sponge construction and uses two-stage absorbing and squeezing. Since 2007, NIST has adopted KECCAK because of the practical attacks on SHA0, SHA1, and SHA2. KECCAK became the rival standard in 2015 [47]. However, some research [48,49] has indicated that SHA3 can suffer from fault injection threats.

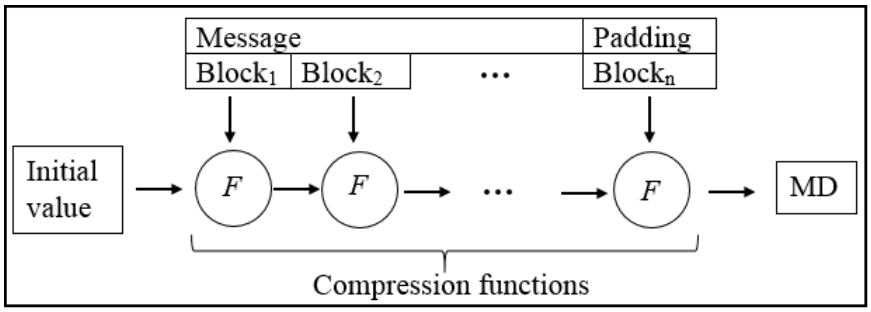

Figure 2. The Merkle–Damgard construction of SHA (0, 1 and 2) hash functions.

SHA is a one-way function consisting of two phases that divide the message into blocks of the same size (such as 512 or 1024). A set of zeros is added and followed by one at the end of the last block of the message [18]. This phase is called preprocessing or padding. The second stage is the MD computation. At this stage, all message blocks are entered into the iterations (SHA1 (80), SHA2 (64), and SHA3 (256)) one by one, containing constants and logic operations (OR, AND, and XOR) in the compression function (F) to produce MD. Each hash algorithm produces a fixed length of MD such as 160 for SHA1 and 224, 256, 384, and 512 for SHA2 and SHA3 [50]. Table 1 shows the comparison between SHA1, SHA2, and SHA3 [51]. Many existing schemes to collect data in WSN [18,52–54] have used the SHA algorithm to support integrity and authentication. However, these schemes do not address the collision and preimage problems in the SHA algorithm [31–35].

Table 1. Comparison of SHA family.

Algorithm	**SHA1**	**SHA2**				**SHA3**			
MD	160	224	256	384	512	224	256	384	512
Word size	32	32	32	64	64	64	64	64	64
Block size	512	512	512	1024	1024	1152	1088	832	576
Message size	$<2^{64}$	$<2^{64}$	$<2^{64}$	$<2^{128}$	$<2^{128}$	-	-	-	-
Iterations	80	64	64	80	80	24	24	24	24
Security	80	112	128	192	256	112	128	192	256
Weak security	Yes, practical such as Collision and preimage	Yes, practical such as preimage and length extension				Yes, theoretical such as fault injection			
Performance	Fast	Less				Lowest			
Year	1995	2004				2015			
Designer	NSA					Guido Bertoni and et al.			
Construction	Merkle–Damgård					Sponge			

– **BLAKE family**

Aumasson et al. [55] proposed a BLAKE hash algorithm to overcome the efficiency problems in previous hash algorithms. This algorithm offers several features such as simplicity, speed, and parallel operations in hardware and software implementations. It is immune to second preimage, side-channel and length-extension attacks. BLAKE implements HAIFA construction which is an enhanced version of Merkle–Damgård. This development of construction is accomplished by adding a salt and a counter to the algorithm stages to prevent security vulnerability for second preimage attacks in Merkle–Damgård. BLAKE's local wide-pipe structure also makes collision attacks impossible [47]. BLAKE uses the LAKE hash algorithm and compresses the message blocks in hash-tree constructions with Bernstein's stream cipher ChaCha, which is a variation of Salsa20-256. Skein and Grøstl [56] considered NIST, a BLAKE

of competing algorithms, in the final round of hash algorithms such as KECCAK. BLAKE supports several versions: 244, 256, 384, and 512. Subsequently, Aumasson et al. [57] developed BLAKE2 to improve the speed in software implementation and to reduce memory. BLAKE2 has 32% less memory than BLAKE. In addition, BLAKE2 contains two versions, BLAKE2s and BLAKE2b, to be used with 32-bit and 64-bit platforms, respectively. Moreover, the authors developed the latest versions, BLAKE2bp and BLAKE2sp, to improve the speed of MD production during parallel processes. Table 2 shows the BLAKE family [50].

Table 2. Versions of BLAKE hash function.

BLAKE Version	Word	Message	Block	MD	Salt	Round
BLAKE-28	32 bits	$<2^{64}$	512	224	128	14
BLAKE-32	32 bits	$<2^{64}$	512	256	128	14
BLAKE-48	64 bits	$<2^{128}$	1024	384	256	16
BLAKE-64	64 bits	$<2^{128}$	1024	512	256	16
BLAKE2s/BLAKE2sp	32 bits	-	256	128-256	64	10
BLAKE2b/BLAKE2bp	64 bits	-	512	160-512	128	12

Figure 3 shows the architecture of the BLAKE hash function. In BLAKE, the message is divided into blocks, and the last block is padded with 1 followed by zeros to complete the last block size to 512 or 1024 bits. BLAKE consists of two parts: the compression function and iteration mode. The compression function consists of chain value, message blocks, salt value, and counter value. The BLAKE compression function uses three phases: initialization, round functions, and finalization. The initialization phase uses the chain value, salt, and counter to create a 4×4 matrix, and produces a 16-word value for different initializing states (V). These states are entered into the round function (r) with parallel rounds in the phase of the round functions. The output of this phase is a new V that is used to generate the chain value for the finalization phase. In the finalization phase of the chain, salt and new state values are applied with $\oplus$ operations to produce a new chain value. BLAKE is one of the fastest hash algorithms and has strong security [50,58]. Recent research has pointed out that BLAKE is a suitable algorithm for source limited devices [59,60].

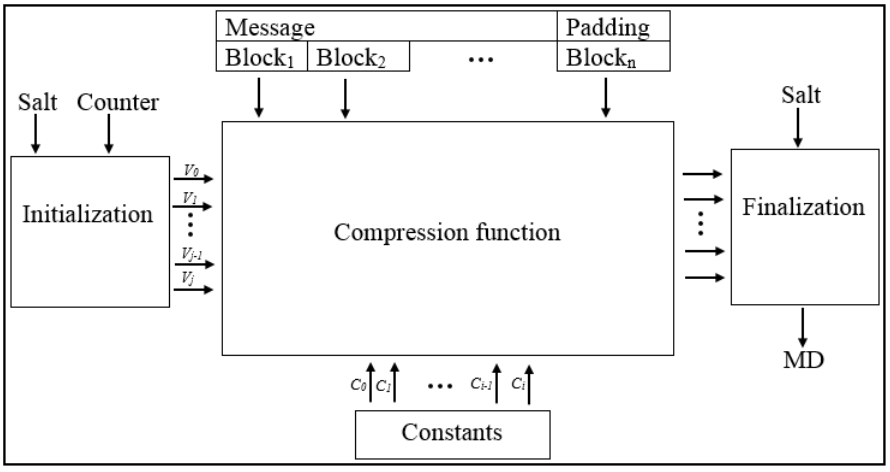

Figure 3. Architecture of BLAKE hash function.

- **De-identification mechanism**

  Encryption and k-anonymity mechanisms are applied to hide patients' data. However, these mechanisms suffer from serious-shortcomings. For instance, encryption of collected data [61] has the following drawbacks:

1. A temporary HC provider such as a researcher doctor will not benefit from the encrypted data, and, if he/she is able to get the collected data by the decryption process, this is a security weakness in the HWSN system.
2. Huge datasets encryption is dramatically burdening for the *LS* system, which causes complexity of operations and processor power consumption [62].
3. The datasets of collected data perform intensive and continuing operations on medical records such as add, delete, and edit, and, if the records are encrypted, this will multiply the burden on the *LS* [63].
4. Encryption can contain implicitly direct information about the HC patients. The breach of this encryption will expose the patients' information to intruders [64].

The k-anonymity of collected data suffers from the following:

1. The removal process of all the patients' information obstructs the HC provider from dealing with the linked patients' data [61].
2. Inserting a large set of false medical records would greatly reduplicate the dataset size. Consequently, this process consumes *LS* resources, particularly with the intensive and continuous access of the datasets by HC providers.

To address these disadvantages, we use random pseudonyms in REISCH's requests to hide the correlation of patients' information with data. The medical records transmitted/stored among the sensors, *CH*s and *LS*, do not contain any patients' real information. This mechanism prevents the intruders from identifying patients' IDs. In addition, this mechanism is fast and does not need complex operations. When the EMR system wants to add a new HC provider/patient, the REISCH sends a request to the remote servers (*CS* and *AS*), which provides *LS* with the required information about updating random pseudonyms. These random pseudonyms are linked with the users' IDs. This mechanism enables sensors to access and store a specific patients' data without exceeding granted privileges.

- **One time passcode (OTP)**

  OTP is a forceful way of validating sensors in HWSN environments if applied with reliable signature technologies. Using a static passcode/nonce without other validation mechanisms is a security weakness with respect to attacks. Thereupon, OTP presents significant support to the validation process. This mechanism is a countermeasure against replay, MITM, and DoS threats [65]. The intruder cannot utilize this passcode/nonce to authenticate the HWSN later. The sensor sends OTP within a validation request. If the validation process is achieved, the *LS* will delete OTP from memory and it will be unacceptable to use it again. OTP provides a strong mechanism to relieve the intruders' risks in the HWSN communications. In REISCH, we apply OTP to get a random nonce with each link to sensors in HC applications to guarantee that only legitimate sensors are communicated to the HWSN.

- **Efficient HC data management using XML**

  The other important part of the proposed EMR system is the repository. The repositories contain data in various contexts since these systems have difficulties dealing with these different coordinates for data. The extensible access control (XML) is considered convenient for the exchange of various data via different environments. XML is the symbolic, simple, and flexible language designed to manage, describe, and exchange data across the Internet. It divides the data into the form of useful information through data organization, for the purpose of sharing data across different systems and storing them in the dataset. Moreover, XML has several features that make it suitable for data management, such as support for unicode, the representation of computer data structures (trees, records, and lists), and using a formula read by both humans and computer. However, XML should support the security mechanisms to provide different levels of protection of

sensitive data in the whole or part of the XML document [66]. Access to the data is a challenge in big data management systems that use different techniques. In addition, the exchange of information over an insecure environment has become essential, particularly in HC applications. However, this information needs mechanisms to identify the arrival of unauthorized users to protect patient data. Patient data transmitted between sensors (nodes and CHs) and network devices (such as a nurse and a LS device) need data management algorithms to maintain both performance and security at the same time. EMR including patients' confidential data and private information needs to be accessed by HC professionals. Thus, sharing such EMR without breaching a patient's privacy requires EMR management in an efficient and secure manner. XML technology has begun showing its superiority in the exchanging of complex data over different systems.

- **Homomorphic scheme**

  A homomorphic is a mechanism for merging all messages and signatures together to improve both performance efficiency and security. This mechanism consists of many types such as linear, polynomial, full, and aggregate signature [67]. In this study, we focus on the aggregate signature because it deals with multi-sensors signatures, messages, and different private keys depending on different devices such as sensors. Furthermore, this process is extremely suitable for multihop-based networks during the integration of signatures into a single signature. We assume that we have a range of messages $M = \{m_1, \ldots, m_n\}$ and a range of signatures $S = \{s_1, \ldots, s_n\}$, M contains all group's messages together, where S is one signature for all signatures, A is an aggregate function, and V is a verification function. The process of homomorphic signatures is as follows:

  - Each device generates K_{pr} and K_{pu} keys and broadcasts the K_{pu} keys to network members.
  - Each device signs the m by the signature algorithm, which includes the device's ID, message and private key $s(K_{pr}, m_i, ID)$.
  - The aggregation procedure in the intermediate nodes such as CH relies on A to collect all public keys, messages and signatures $A(K_{pu_1}, \ldots, K_{pu_n}; m_1, \ldots, m_n; s_1, \ldots s_n)$.
  - The verification procedure will be in the final entity such as LS, which uses V to validate the signatures $V(K_{pu_1}, \ldots, K_{pu_n}; m_1, \ldots, m_n; s_1, \ldots s_n)$. If the verification process fails, it means that the data integrity operation is incorrect.

  The homomorphic aggregate signature scheme is important to support the performance of network devices by making the intermediate nodes such as CH perform a single signature process for all members' signatures of the group rather than the signature verification process (the ECDSA verification process consumes more time and energy than the signature process) [68]. In addition, homomorphic increases security measures in preventing the tracking of patients' information and data or changing signatures of legitimate network devices [69].

4. The Proposed Data Collection Scheme

In this section, we provide details about REISCH that possesses security and performance efficiency features in HWSN. The section consists of three parts: the network model, security goals, and proposed data collection protocols.

4.1. Network Model

Figure 4 shows the network model in which our proposed REISCH scheme is based:

1. Sensor (SN): This entity collects raw data related to a specific patient. It sends these data to the CH.
2. Cluster Head (CH): This entity aggregates data from the sensors that followed it. Then, it sends these data to the LS.
3. Local Server (LS): This entity receives data from all CHs in each round and stores it in EMR's repository. These data are subsequently sent periodically to the Central Server (CS).

4. Central Server (*CS*): This entity is a gateway accessing remote servers such as the Attributes Server (*AS*) and the Data Server (*DS*). It receives data from the *LS* and sends data to the *DS* after being authenticated by the *AS*. Security procedures in *AS* and *DS* are left for future directions.

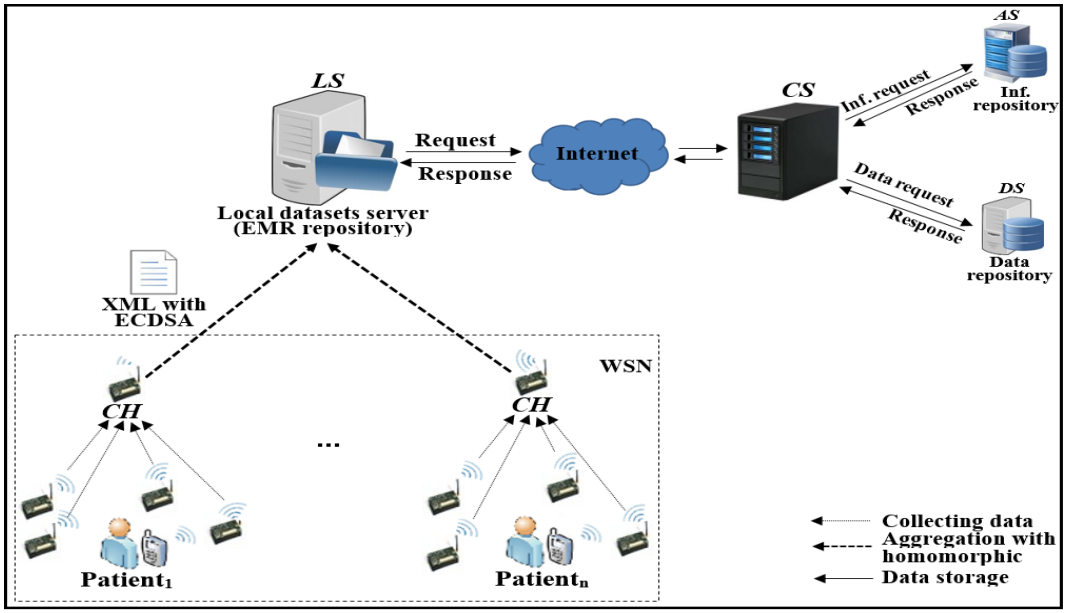

Figure 4. General REISCH model.

Among the WSNs, the low-energy adaptive clustering hierarchy (LEACH) protocol is used. LEACH uses clustering architecture to improve the WSN lifetime. More details about this protocol are available in [70]. Each group of *SN*s collects raw data for a specific patient [71]. These *SN*s sign data before sending them to *CH*s. Each *CH* aggregates data and signatures from his followers. Then, each *CH* uses homomorphic property with all data and signatures without verifying the signatures to reduce energy consumption on the *CH* and send them to the *LS*. As the *LS* has unlimited resources, it verifies and validates collected data from *SN*s. The *LS* sends data stored on the EMR's repository to the central repository to allow HC users (patients and providers) to access them by sending authentication/authorization requests to the *CS*, *AS*, and *DS*. This paper focuses on performance and security issues in *SN*s, *CH*s, *LS*, and *CS*. Security issues for datasets and transferred data in *CS*, *AS*, and *DS* are left for future works.

4.2. Security Goals of REISCH Scheme

The REISCH has the following security services:

- **Information confidentiality**: This service is achieved to hide *SN*s/patients' identities and to protect patients secrecy from disclosure by intruders.
- **Data integrity**: This service is required to protect the patient data from tampering by intruders. The collected data should arrive at the intended target without alteration to provide a reliable communication channel among *SN*s, *CH*s, *LS*, and *CS* [72].
- **Non-repudiation**: This is a feature to prove that the *m* is sent by a particular *SN* in the HWSN. If a legitimate entity in HWSN performs internal attacks, he/she cannot deny his/her messages while availing the privileges granted to him/her.
- **Forward secrecy/Backward secrecy**: This requirement is performed when network entities use new keys and parameters temporarily without depending on old ones in the future. While backward secrecy prevents the newly joined sensors from accessing previous messages before entering the HWSN.

- **Freshness**: It indicates that the data collection message is new and updated to guarantee that the intruder cannot replay the previous message at a later time. This goal is accomplished by a checking of time, a random passcode, and random signatures within each data collection round to counteract spoofing risks such as replay, MITM, and impersonation.
- **Security of Localization**: This feature ensures that the patient/sensor's real location is protected from detection, or sends error messages to the *LS* by an intruder.
- **Scalability**: HWSN applications elaborate in a scalable environment in both data and devices. Thus, these applications need data collection schemes capable of processing and adapting to the ever-increasing number of devices of the HWSN. This feature indicates the ability of the data collection scheme to properly handle huge HWSN devices. Public key signature schemes are convenient to provide this requirement [73].
- **Survivability**: It provides a certain level of services in patient data collection or network capability to withstand failure/threats in an appropriate manner and continue to provide services between *SN*s and *LS* for as long as possible.
- **Accountability**: This property means tracking the behaviour of malicious threats/suspicious activities by legitimate users/counterfeiting attacks in accessing EMR repository.
- **Efficiency**: HWSN sources such as energy, storage, and processor should be within the design objectives of security protocols in HWSN.

4.3. REISCH's Scheme

In this subsection, we explain the details of REISCH in terms of entities preparation, using ECDSA-BLAKE2bp, applying a camouflage signature, and implementing homomorphic and REISCH protocols.

4.3.1. Entities Preparation

To start collection and storage processes, the HWSN network should be prepared with the following points:

- Each sensor (SN_i) and *LS* server provides SN_i pseudonym (SN_{Pseud}), SN_i pseudonym signature ($SigLS_i(SN_{Pseud})$) and SN_i location (SN_{SL}).
- All entities (SN_i, CH_i, *LS and CS*) generate K_{pu_i} and K_{pr_i} to apply asymmetric cryptographic.
- Each entity broadcasts K_{pu_i} to network members.
- Each SN_i uses ECDSA signatures (*SigSN* and *SigCH*) to achieve collected data integrity.
- Each server (*LS* and *CS*) uses ECDSA signatures (*SigLS* and *SigCS*) to achieve storage data integrity.

4.3.2. Using ECDSA-BLAKE2bp

REISCH implements ECDSA-BLAKE2bp (NIST prime 256-bit) to sign all messages (m) among HWSN entities (SN_i, CH_i, *LS and CS*). The collected data are formatted as XML-enabled files to allow different devices in the HWSN network to deal easily and flexibly with these records. We use the BLAKE2bp algorithm instead of SHA1 to perform the hash function on collected and stored data (Section 3.3; in the second point in both signature generation and signature verification, we use BLAKE2bp (d) instead of SHA1 (d)). In REISCH, we use ECDSA-BLAKE2bp to ensure data integrity as well as add SN_{Pseud} within *SigSN* to prevent changing data. *LS* and *CS* accept only valid signature after verification. The high performance and security of the ECDSA-BLAKE2bp algorithm makes it an appropriate choice to protect EMR health records. Using ECDSA-BLAKE2bp with XML also adds the feature of managing medical records in HWSN.

4.3.3. Applying Camouflage Signature

REISCH uses the camouflage process to hide the data signature and completely prevent traceability, analysis or alteration of data. The camouflage process starts by signing the data to obtain a 64-bit MD

and then adding a 64-bit counterfeit signature to a total length of 64-bit + 64-bit = 128-bit. In addition, each SN_i adds padding (0000) to become the total length of the 132-bit signature, as shown in Figure 5. SN_i performs the process of exchanging data signature segments based on Parity (even/odd) value. It receives this value invisibly from the LS because this value is included in the ephemeral random value ($SigLsE_i$). SN_i tests $SigLsE_i$; if "even", it divides the 132-bit into four segments (each segment to 33-bit) and exchanges the segments. Then, SN_i truncates the 32 bit from the first segment and divides it into four segments (each segment to 8-bit). If $SigLsEi$ is "odd", it divides the 132 bit into three segments (each segment to 44-bit) and then exchanges the segments. It then truncates the 42 -bit and divides the first segment into three segments (each segment to 14 bit). Because the exchanging operation is based on Parity sent from the LS, this process prevents the detection of the original signature of the data and prevents the data from being changed. Thus, this process protects patient data from tampering or alteration.

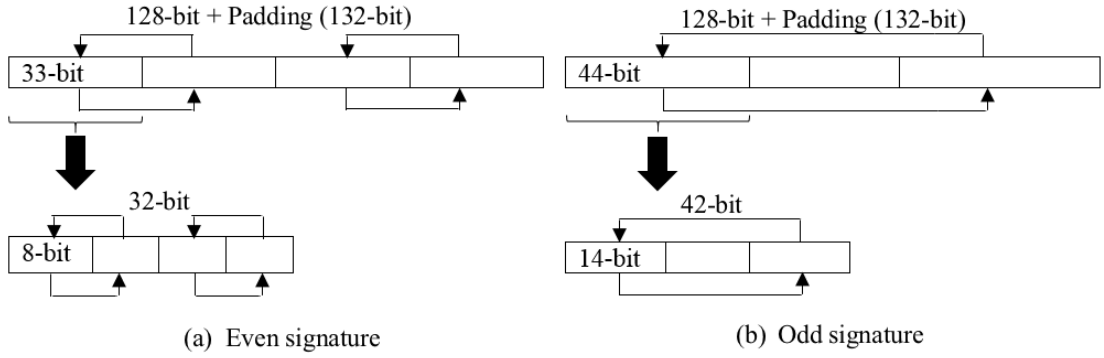

Figure 5. Camouflage signature.

4.3.4. Implementing Homomorphic

REISCH uses the homomorphic property with the ECDSA-BLAKE2bp algorithm to increase network performance. Because the verification process in ECDSA consumes more time and processing than the signature process, it is convenient to use the homomorphic property in HWSN to support both performance and security. The LEACH protocol is based on the principle of clustering to reduce energy consumption, thus REISCH uses the aggregate signature to allow CH_i to aggregate signatures and data without using verification. To double security in REISCH, CH_i performs the process of aggregating temporary signatures such as $SigSnT_{3s}$ and $SigSnT_{4s}$ in addition to random numbers (SN_{RNs}) and data. Temporary signatures contain unclear original signatures that prevent an intruder from penetrating patient data. The homomorphic procedure reduces energy consumption and thus increases the possibility of using the ECDSA algorithm with HWSN for as long as possible.

4.3.5. REISCH's Protocols

The REISCH scheme consists of three protocols. During these protocols, REISCH provides reliable data collection processes to protect collected patients' data.

Protocol 1 between SNs and CHs:

This protocol performs the data collection process (Figure 6 shows the first protocol processes between SN_i and CH_i in the data collection). The process is as follows:

SN_i Side

- At the beginning of each round, each SN_i receives a one-time passcode (LS_{OTP_i}) and a random number (SN_{RN_i}). This LS_{OTP_i} contains an ephemeral random value ($SigLsE_i$) of the same length as the signature. SN_i extracts $SigLS_i(SN_{Pseud})$ from the dataset and executes $\oplus$ to extract the secret value $SigLsEi$.

- Then, SN_i executes the *Parity* (as shown in Section 4.3.3) process based on $SigLsEi$ to get the temporary signature ($SigSnT_1$).
- After that, SN_i generates an ephemeral random value ($SigSnE_i$) with the same signature length and uses it with $SigSnT_1$ to compute the $SigSnT_2$ value.
- Next, SN_i computes the Dif value that represents the subtraction value of the distance between CH_i and SN_i (S_NC_HD) and the distance between LS and SN_i (S_NL_SD). Dif specifies that SN_i is within the HWSN framework (1000 m $\times$ 1000 m).
- Additionally, SN_i computes a new timestamp (SN_{TS}) and one time passcode (SN_{OTP}). SN_i also performs a hidden process for SN_{TS} and SN_{OTP} at a temporary value (SN_{TS_t}) with the addition of a value of only seconds (SS) at the end of the SN_{TS_t}.
- Furthermore, SN_i uses SN_P to concatenate secret parameters such as SN_{TS_t}, SN_{OTP}, SN_{RN_i}, SN_{Pseud} and SN_{SL} to match them at the LS. To protect both SN_P and $SigSnE_i$, SN_i uses the $\oplus$ operation to hide them by calculating the temporary values of $SigSnT_3$ and $SigSnT_4$. At this point, SN_i computes the message (SN_m) and sends it to CH_i which is a sequence of $SigSnT_3$, $SigSnT_4$, SN_{RN_i}, Dif, SN_{TS_t} and data collection.

CH_i Side

- CH_i also receives LS_{OTP_i} of the LS and SN_m of SN_i.
- Afterwards, CH_i truncates Dif_i and tests its value within the HWSN framework by computation $Dif_i \leq$ Maximum value, where the Maximum value should be less than or equal to 707.1068.
- Then, CH_i computes the timestamp (CH_{TS_1}) to prevent late messages.
- CH_i truncates SS from SN_{TS_t} to obtain SN_{TS_i}. If the difference between CH_{TS_1} and SN_{TS_i} is less than the $\triangle T$ delay rate (we assumed that $\triangle T = 3$), namely, that the message is fresh.

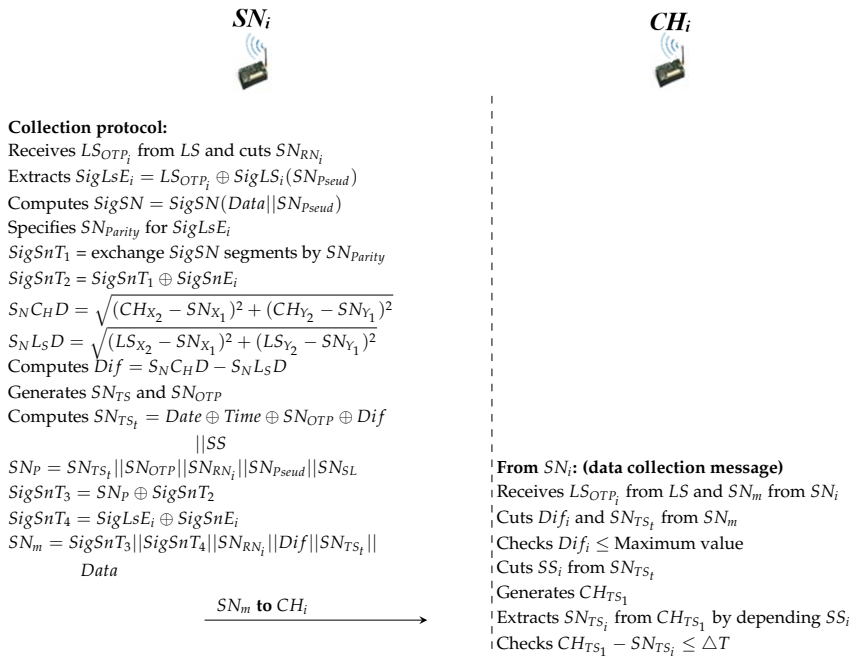

Figure 6. Data collection protocol.

Protocol 2 between CHs and LS:

This protocol performs the data aggregation process (Figure 7 shows the second protocol processes between CH_i and LS in the data aggregation). The process is as follows:

CH_i Side

- Each CH_i receives temporary signatures, random numbers and collected data from its SN_i followers.
- Then, CH_i executes the signature process $SigCH$ for the temporary signatures received ($SigSnT_{3_s}$) of its SN_i followers.
- Thereafter, CH_i extracts the $SigLsE_i$ unique value from LS_{OTP_i} similar to the first protocol based on $SigLS_i(CH_{Pseud})$ stored.
- Next, CH_i Performs CH_{Parity} process based on $SigLsE_i$ extracted (as described in Section 4.3.3) to compute $SigChT_1$. Moreover, CH_i computes $SigChT_2$ depending on the $SigChT_1 \oplus SigLsE_i$ operation.
- After that, CH_i generates CH_{TS_2} and CH_{OTP} to prevent the problem of replaying messages later. CH_i calculates CH_P which represents the sequence of secret parameters. In addition, CH_i computes CH_A to complete the process of aggregating temporary signatures ($SigSnT_{3s}$ and $SigSnT_{4s}$), random numbers (SN_{RNs}), and collected data ($Data_s$).
- Finally, CH_i computes CH_m and sends it to the LS.

LS Side

- After LS sends LS_{OTP_i} for all SN_i, it waits to receive CH_m of all CH_i per round. The LS truncates CH_{RN_i}, SS_i and CH_A from each CH_m received. It uses SS_i to reconfigure CH_{TS_2}; in addition, the LS generates a timestamp (LS_{TS_1}) and tests $\triangle T$ between LS_{TS_1} and CH_{TS_2} to confirm the freshness of the message.
- Then, it tests whether CH_{RN_i} matches the value previously sent. If CH_{RN_i} is correct, it is used to determine CH_{Pseudi} and the latter is used to determine CH_i location (CH_{SL_i}).
- Afterwards, the LS retrieves temporary signatures and random numbers ($SigSnT_{3_s}$, $SigSnT_{4_s}$ and SN_{RN_s}) from CH_A. The LS uses the $SigLsE_i$ value to specify a $Parity$ (even/odd) value for all SN_i and CH_i. It computes a signature ($SigLS_{1_i}$) for all SN_i signatures that followed a specific CH_i ($SigSnT_{3_s}$) and exchange the $SigLS_{1_i}$ segments based on CH_{Parity}.
- After that, the LS calculates $SigLsT_{1_i}$ which equals $SigChT_2$ in CH_i based on $SigLS_{1_i} \oplus SigLsE_i$. To ensure the legitimacy of CH_i, the LS extracts the secret parameters at CH_{P_i} and tests the match CH_{Pseudi} and CH_{SL_i} in the datasets.
- At this point, the LS checks for data integrity collected by SN_i. Similarly, the LS uses SN_{RN_i} to determine SN_{Pseudi}, and performs data signature ($SigLS_{2_i}$) that equals the $SigSN$ in SN_i and exchanges the $SigLS_{2_i}$ segments based on SN_{Parity_i}.
- Next, the LS uses $SigSnT_{4_i}$ and $SigLsE_i$ to extract $SigSnE_i$. Thereafter, the LS uses $SigSnT_{3_i}$ and $SigSnE_i$ to compute $SigLsT_{2_i}$.
- Finally, the LS extracts the secret parameters for SN_i from SN_{P_i} and tests matching SN_{Pseudi} and SN_{SL_i} in datasets. If all signatures and parameters are validated correctly, then that the data collected by SN_i are legitimate and correct and have not been tampered with by the intruder.

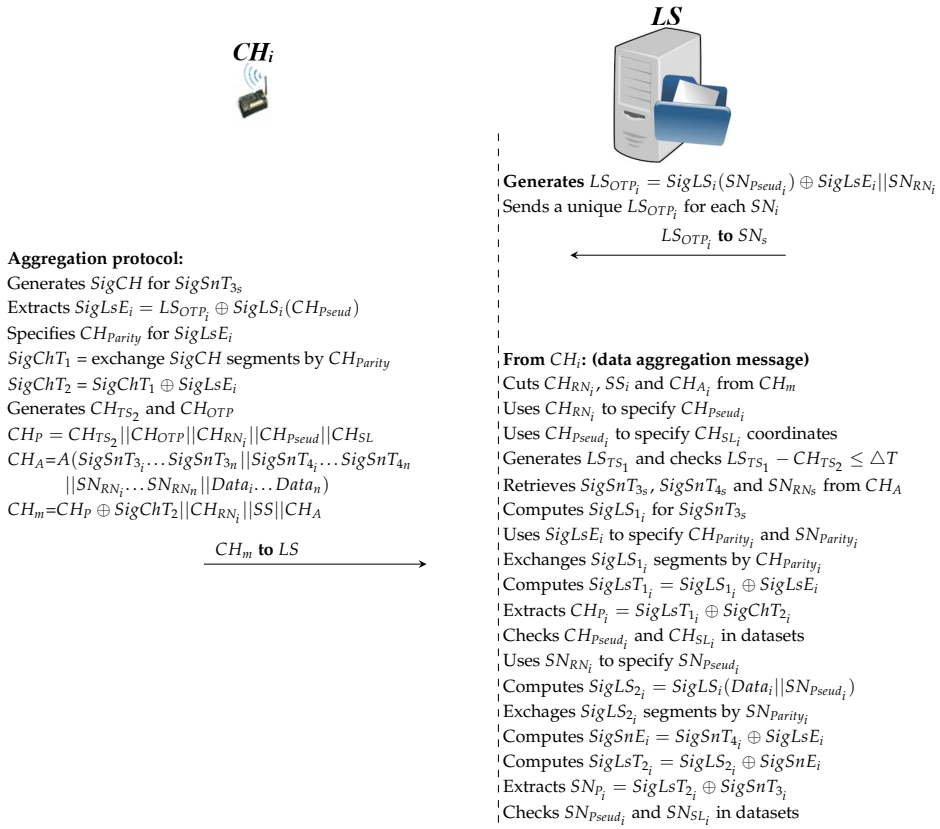

Figure 7. Data aggregation protocol.

Protocol 3 between *LS* and *CS*:

This protocol performs the data storage process (Figure 8 shows the third protocol processes between the *LS* and *CS* in the data storage). The process is as follows:

LS Side

- In sending case to *CS*, the *LS* initially generates a new pseudonym (LS_{Pseud_n}) and timestamp (LS_{TS_2}) to prepare for the process of sending data to the *CS*.
- Then, the *LS* computes the *SigLS* signature based on the *CS* 's old pseudonym (CS_{Pseud_o}).
- After that, the *LS* generates and sends LS_{OTP} to the *CS*, which is based on the *SigLS*, LS_{Pseud_n}, LS_{TS_2} as well as appending *SS* at the end of LS_{OTP}.
- In receiving case from *CS*, *LS* truncates parameters embedded within CS_m. Thereafter, the *LS* generates LS_{TS_3} to check the arrival time of CS_m.
- Furthermore, the *LS* computes CS_{OTP} that relying mainly on LS_{Pseud_n}. Afterwards, the *LS* extracts CS_{Pseud_n} to calculate $SigLS_3$. The *LS* tests matching $SigLS_3$ and *SigCS*, and if the result is identical, this means that mutual authentication process between the *LS* and *CS* is performed correctly and legitimately.
- After this stage, the *LS* prepares the data storage request to *CS*. First, the *LS* generates LS_{TS_4} and LS_{RN} to ensure randomness and freshness.
- After that, the *LS* computes the $SigLS_4$ signature that depends on the LsT_1 temporary parameters.
- Then, the *LS* computes the $SigLS_5$ data signature that depends on temporary parameters such as LsT_2, LsT_3, and $SigLS_4$ as well as the *Data*.
- Finally, the *LS* sends LS_m which includes $SigLS_5$, *SS*, LS_{RN} and *Data* to *CS*.

CS Side

- In sending case to *LS*, *CS* generates CS_{TS_1}, CS_{Pseud_n}, CS_{OTP}, and CS_{RN}. *CS* uses CS_{TS_1} to test the message arrival time of the *LS*. Depending on generated secret parameters, such as CS_{OTP}, *CS* computes CsT_1 and CsT_2 temporarily.
- In addition, the *CS* generates a *SigCS* signature that includes the temporary value (CsT_2).
- At this point, the *CS* computes and sends CS_m to *LS* containing the sequence of parameters such as *SigCS*, CsT_1, CsT_2, *SS* and CS_{RN}.
- In receiving case from *LS*, *CS* receives LS_m of *LS*. The *CS* generates CS_{TS_2} new to test access time LS_m. The *CS* calculates $SigCS_1$ and $SigCS_2$ similarly to $SigLS_4$ and $SigLS_5$ respectively.
- At this point, the *CS* checks matching $SigCS_2$ and $SigLS_5$, and, if the result is identical, it means that *CS* received patients' data from the *LS* correctly and integrated without any changes by malicious attacks.

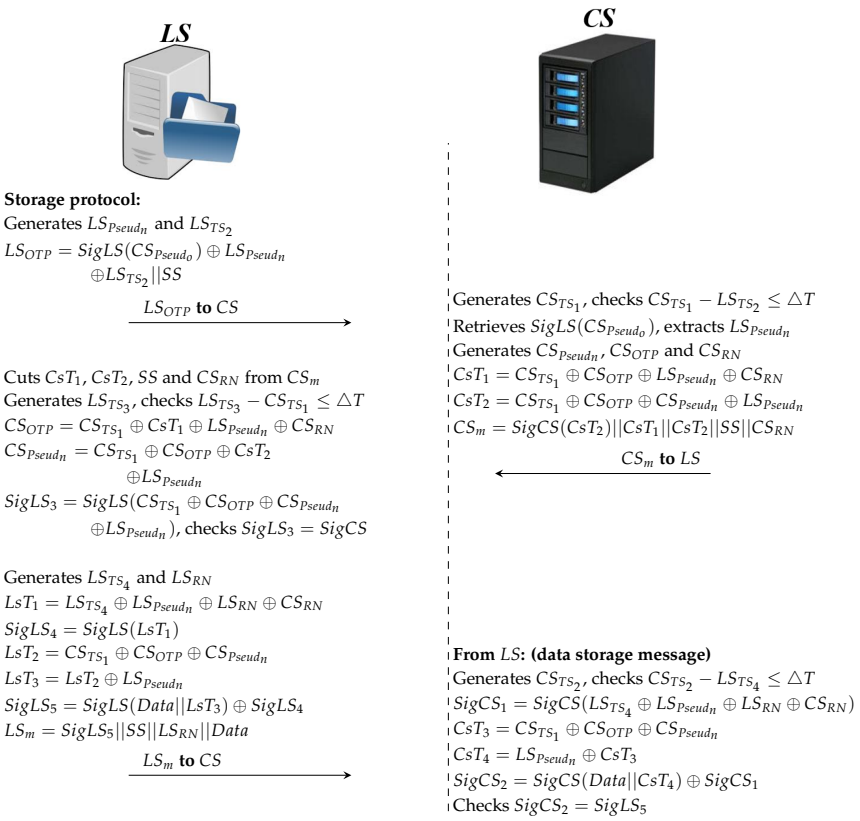

Figure 8. Data storage protocol.

5. Discussion

In this section, we discuss the security and performance analysis for the REISCH scheme. Analyses demonstrate that REISCH is efficient for use in patient data collection within the HWSN environment in terms of security and performance.

5.1. Security Analysis

In this section, the theoretical and experimental security analysis is provided to examine REISCH protocols in repelling known attacks.

5.1.1. Theoretical Analysis

In this section, we examine the REISCH scheme theoretically with the set of threats mentioned in the threat model. We provide a theoretical analysis of REISCH resistance to known attacks as follows:

- **MITM and replay**

  Proof 1: An intruder tries to change or delete part of data/information when transferred between the network's entities. This situation is not possible because REISCH applies the ECDSA algorithm to sign data as well as some information such as SN_{Pseud}. Additionally, an intruder cannot replay a message late due to the REISCH's entities use of timestamps such as SN_{TS} and CH_{TS}. Consequently, REISCH resists MITM and replay attacks successfully. □

- **DoS**

  Proof 2: An intruder applies a DoS attack to destroy availability service in servers such as the LS and CS. The servers in REISCH initially check lightweight parameters such as $SigLsE_i$ in LS and CS_{Pseud_o} in CS before completion of the authentication process. Moreover, these parameters change randomly in the communication process between entities. This procedure allows servers to check small parameters and prevent DoS duplicate messages. Therefore, REISCH withstands DoS threats. □

- **Localization**

  Proof 3: An intruder tries to use the Sybil attack by using many legitimate SN IDs with fake data. Since SN_i waits for random $SigLsE_i$ from LS each round, the intruder cannot deceive LS with fake data. Additionally, an intruder uses Wormhole attack by using many SN_i to camouflage communications between network entities. Each SN_i sends implicitly SN_{SL_i} to LS and Dif to CH_i as well as a timestamp. These parameters prevent counterfeit communications. Furthermore, if an intruder aims to apply a Sinkhole attack using node as a sink to attract all patient data from SN_i, it cannot apply to REISCH because LS sends a unique LS_{OTP_i} including $SigLS_i(SN_{Pseud_i})$ for all SN_i. That intruder fails to detect $SigLS_i$ and SN_{Pseud_i}. Hence, REISCH strongly overcomes localization attacks. □

- **EMR repository**

  Proof 4: Assume that an intruder can penetrate datasets in LS. First, LS does not contain real patient information (real information such as the name is stored in AS). When the intruder gets these data, he/she cannot disclose that they belong to a particular patient. Second, the LS 'datasets tremendously are difficult to penetrate. Furthermore, LS contains partial data for patients because the total data and patient history are transferred to DS by CS periodically. Thereupon, REISCH resists the EMR repository attack. □

- **Eavesdropping**

  Proof 5: When an intruder eavesdrops and gets some of the messages transferred among SN_i, CH_i, LS, and CS, this intruder will not benefit from these messages that are being trapped because these messages contain no real information. Furthermore, the secret parameters, are completely hidden. Thus, REISCH prevents eavesdropping attacks from revealing patient information. □

- **Node replication**

  Proof 6: An intruder applies a node replication attack using more than one SN_i with the the same legitimate ID. In REISCH, we suppose that all SN_i are inside a specific area in the hospital or clinic. Therefore, any SN_i outside this area finds it extremely difficult to send messages from fake SN_i with same legitimate ID. In addition, LS waits SN_m by CH_i at the same number of SN_i and the LS removes replicated SN_m or $SigSnT_3$. In addition, when SN_i dies, LS records this situation in the dataset to prevent replication risks. As a result, REISCH effectively resists replication attacks. □

- **Collision and preimage**

  Proof 7: An intruder tries to implement a collision (the generation of two different messages that produce the same MD =h (m) = h (m')), preimage (the generation of a message that produces the same existing MD value as h (m) = MD), and second preimage (the generation of a different message from the received message and produce the same existing MD value) attacks when messages and signatures are transferred between REISCH's entities. These attacks cannot be

implemented on REISCH protocols because our protocols use the BLAKE2bp hash instead of SHA1, which resists these attacks. Consequently, REISCH successfully prevents collision and preimage attacks. □

5.1.2. Experimental Analysis

In this section, we use the AVISPA tool to simulate the REISCH's protocols. This tool is extremely important to examine/check applicability passive and active attacks on security protocols. We tested the exchanging of SNs' data/information with network entities (CH_i, LS and CS) and analyzed the results, as shown following subsections.

AVISPA Summary

AVISPA is a formal verification and validation tool that is used to trace and analyze threats on HWSN's security protocols. This tool depends on high-level protocol specification language (HLPSL) to achieve its functions. In addition, AVISPA includes backends to trace/detect attacks in many ways, intermediate format (IF) to read HLPSL's codes and output format (OF) to produce simulation results. In this paper, we depend on the On-the-Fly Model-Checker (OFMC) and the Constraint-Logic-based Attack Searcher (CL-AtSe) backends because our scheme deals with XoR operations. It presents a simple and easy way (push-button) to run HLPSL codes (the readers can get more information about AVISPA details in [72,74]). Additionally, the communication channel in AVISPA is Dolev–Yao (dy) that is used to transfer the sensors' data/information during HWSN's simulation. Moreover, AVISPA has been used in recent research because this tool has significant advantages such as threats tracking, implementation robustness, simplicity, analysis of results and statistics [15,72,75,76].

REISCH Scheme with AVISPA

In this subsection, we explain the REISCH scheme in AVISPA. REISCH includes four roles, namely localServer (LS)), sensori (SN_i), clusterHeadi (CH_i), and centralServer (CS), as well as supporting roles, namely session and environment. Moreover, there are three sections to complete communication properly and securely: transition, composition, and goal specification. The transition section is used in the essential roles to keep a correct communication sequence. The composition section is used in the supporting roles to connect essential roles in specific sessions. The goal specification section includes security goals such as secrecy and authentication. Secrecy means known secrets only for specific entities while authentication depends on witness (freshness claim) and request (validation) processes to perform strong authentication. In addition, our scheme uses parameters such as RCV (receiving process), SND (sending process), _inv (private key), dy (communication channel by Dolev-Yao model), and *intruder_knowledge* (known information for an intruder). We assume that the intruder uses the public key (k_i) and knows public keys for REISCH entities ($kSNpu$, $kCHpu$, and kLS). Figure 9 shows the REISCH framework in AVISPA. Figures 10–13 show the REISCH roles in AVISPA.

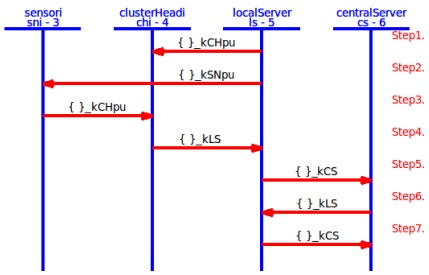

Figure 9. REISCH's framework.

```
role sensori (SNi,CHi,LS:agent, KSNpu,KCHpu,KLSpu:public_key, ECDSA:hash_func, SNpseud,SNsl,Data:text,
            SND,RCV:channel (dy))
played_by SNi def=
local
     State:nat,
     SNrni,SigLsEi,SigSnEi,SNparity:text, MAXv:message,
     SigSN,SigSnT1,SigSnT2,SigSnT3,SigSnT4,SigLSi:text,
     Dif,SNts,SNtst,SNotp,Date,Time,SS,SNp,LSotpi:text
const
     sec1,sec2,sec3,auth3:protocol_id
init
     State := 0
transition
% SNi receives (LSotpi) from LS
1.State=0
  /\RCV (SNi.LSotpi'.SNrni') =|> State':=1
  /\SigSnEi':=new () /\SigLSi':={ECDSA (SNpseud)}_inv (KSNpu) /\SigLsEi':=xor (LSotpi',SigLSi)
  /\SigSN':={ECDSA (Data.SNpseud)}_inv (KLSpu) /\SNparity':=SigSN'/\SigSnT1':=SNparity'
  /\SigSnT2':=xor (SigSnT1',SigSnEi') /\SNts':=new ()/\SNotp':=new ()
  /\SNtst':=xor (Date,xor (Time,xor (SNotp',xor (Dif)))) /\SNp':={ (SNtst'.SNotp'.SNrni'.SNpseud.SNsl)}
  /\SigSnT3':=xor (SNp',SigSnT2')/\SigSnT4':=xor (SigLsEi',SigSnEi')
  /\secret ({SigSN',SigLsEi',SigSnEi'},sec1,{SNi,LS}) /\secret ({SNpseud,SNsl},sec2,{SNi,LS})
  /\secret ({MAXv,SNts'},sec3,{SNi,CHi})
% SNi sends (SNm) to CHi
/\SND (CHi.SigSnT3'.SigSnT4'.SNrni'.Dif.SNtst'.SS.Data) /\witness (SNi,CHi,auth3,{SNtst',SNrni',Dif})
end role
```

Figure 10. SN_i role in HLPSL.

```
role clusterHeadi (CHi,LS,SNi:agent, KCHpu,KLSpu,KSNpu:public_key, ECDSA:hash_func,CHpseud,CHsl:text
                   ,MAXv:message,SND,RCV:channel (dy))
played_by CHi def=
local
     State:nat,
     SNrns,SNrni,SigLsEi,SNparityi,SNts,LSotpi:text, SigCH,SigChT1,SigChT2,SigSnT3s,SigSnT4s,
     SigSnT3i,SigSnT4i,SigLSi,Difi,SNtst,CHts1,CHts2,SS,SSi,CHp,CHparity,CHotp,CHrni,Datai,
     Datas,CHa:text
const
     sec3,sec4,sec5,auth4:protocol_id
init
     State := 0
transition
% CHi receives (LSotpi) from LS
1.State=0 /\RCV (CHi.LSotpi'.CHrni')=|> State':=1

% CHi receives from SNi
2.State=1/\RCV (CHi.SigSnT3i'.SigSnT4i'.SNrni'.Difi'.SNtst'.SSi'.Datai')=|>State':=2
  /\CHts1':=new () /\SigSnT3s':={(SigSnT3s'.SigSnT3i')}/\SigSnT4s':={(SigSnT4s.SigSnT4i')}
  /\SNrns':=SNrni'/\Datas':=Datai' /\SigCH':={ECDSA (SigSnT3s')}_inv (KLSpu)
  /\CHparity':=SigCH' /\SigChT1':=xor (CHparity',CHts1')
  /\SigLSi':={ECDSA (CHpseud)}_inv (KCHpu) /\SigLsEi':=xor (LSotpi,SigLSi')
  /\SigChT2':=xor (SigChT1',SigLsEi') /\CHts2':=new ()/\CHotp':=new ()
  /\secret ({MAXv,SNts},sec3,{CHi,SNi}) /\secret ({SigCH,SigLsEi,CHpseud,CHsl},sec4,{CHi,LS})
  /\secret ({CHotp',CHts2'},sec5,{CHi,LS})
% CHi sends (CHm) to LS
   /\SND (LS. (xor ( (CHts2'.CHotp'.CHrni.CHpseud.CHsl),SigChT2)).CHrni.SS. (SigSnT3s.SigSnT4s
   .SNrns'.Datas')) /\witness (CHi,LS,auth4,{CHts2',CHotp',CHrni,SNrns'})
end role
```

Figure 11. CH_i role in HLPSL.

```
role localServer (LS,CHi,SNi,CS:agent, KLSpu,KCHpu,KSNpu,KCSpu:public_key, ECDSA:hash_func ,SNpseudi
             ,SNsli:text,SND,RCV:channel (dy))
played_by LS def=
local
     State:nat,
     SNrni,SNrns,SigLsEi,SigSnEi,SNparityi,SNp:text,
     SigChT2,SigSnT3i,SigSnT4i,SigSnT3s,SigSnT4s,SigLsT1i,SigLsT2i,SigLS1i,SigLS2i,SigLSi:text,
     CHrni,CHparityi,CHotpi,CHsli,CHts2,SS,Datai,Datas:text,LSpseudo,LSpseudn,CSpseudo,
     CSpseudn:text, LSts2,LSts3,LSts4,LsT1,LsT2,LsT3,LSotpi,LSotpii:text,
     CHpseudi,LSotp,CSotp,CSts1,CsT1,CsT2,CSrn,LSrn,SigLS3,SigLS4,SigLS5:text
const
     sec1,sec2,sec4,sec5,sec6,sec7,auth1,auth2,auth4,auth5,auth6,auth7:protocol_id
init
     State := 0
transition
% Starting signal
1.State=0 /\ RCV (start) =|>
% LS sends  (LSotpi,LSotpii)to SNi & CHi
  State':=1/\ SNrni':=new ()/\SigLsEi':=new ()
  /\LSotpi':=xor ({ECDSA (SNpseudi)}_inv (KSNpu),SigLsEi') /\witness (LS,SNi,auth1,{SigLsEi',SNrni'})
  /\SND (SNi.LSotpi'.SNrni') /\SNrni':=new ()/\SigLsEi':=new ()
  /\LSotpii':=xor ({ECDSA (SNpseudi)}_inv (KCHpu),SigLsEi') /\witness (LS,CHi,auth2,{SigLsEi',SNrni'})
  /\SND (CHi.LSotpii'.SNrni')

2.State=1
% LS receives (CHm) from CHi
  /\RCV (LS. (xor ( (CHts2'.CHotpi'.CHrni'.CHpseudi'.CHsli'),SigChT2')).CHrni'.SS'. (SigSnT3s'.SigSnT4s'
  .SNrns'.Datas')) =|>State':=2 /\SigLS1i':={ECDSA (SigSnT3s')}_inv (KLSpu) /\CHparityi':=SigLS1i'
  /\SigLsT1i':=xor (SigLS1i',SigLsEi) /\SigChT2':=xor ( (CHts2'.CHotpi'.CHrni'.CHpseudi'.CHsli'),SigLsT1i')
  /\SNrni':=SNpseudi/\Datai':=Datas' /\SigLS2i':={ECDSA (Datai.SNpseudi)}_inv (KLSpu)
  /\SNparityi':=SigLS2i' /\SigSnT3i':=SigSnT3s'/\SigSnT4i':=SigSnT4s'
  /\SigSnEi':=xor (SigSnT4i',SigLsEi) /\SigLsT2i':=xor (SNparityi',SigSnEi')
  /\SNp':=xor (SigLsT2i',SigSnT3i') /\request (LS,CHi,auth4,{CHts2,CHotpi',CHrni',SNrns'})
  /\secret ({SigLS2i,SigLsEi,SigSnEi},sec1,{LS,SNi}) /\secret ({SNpseudi,SNsli},sec2,{LS,SNi})
  /\secret ({CHparityi',SigLsEi,CHpseudi,CHsli},sec4,{LS,CHi})
  /\secret ({CHotpi',CHts2'},sec5,{LS,CHi}) /\LSpseudn':=new ()/\LSts2':=new ()
  /\LSotp':=xor (SigLSi,xor (LSpseudn',LSts2'))
% LS sends (LSotp) to CS
  /\SND (CS.LSotp'.SS) /\secret ({LSpseudn,CSpseudn},sec6,{LS,CS})
  /\witness (LS,CS,auth5,{CSpseudo,LSpseudn',LSts2'})

3.State=2
% LS receives (CSm) from CS
/\ RCV (LS.{ECDSA (CsT2')}_inv (KLSpu).CsT1'.CsT2'.SS'.CSrn') =|>
  State':=3/\LSts3':=new () /\CSotp':=xor (CSts1,xor (CsT1',xor (LSpseudn,CSrn')))
  /\CSpseudn':=xor (CSts1,xor (CSotp',xor (CsT2',LSpseudn)))
  /\SigLS3':={ECDSA (xor (CSts1,xor (CSotp',xor (CSpseudn',LSpseudn))))}_inv (KLSpu)
  /\secret ({SigLSi,CSotp'},sec7,{LS,CS}) /\request (LS,CS,auth6,{CSpseudn',LSpseudn,CSts1})
% Prepares message to send data with mutual authentication
 /\LSts4':=new ()/\LSrn':=new () /\LsT1':=xor (LSts4',xor (LSpseudn,xor (LSrn',CSrn')))
 /\SigLS4':={ECDSA (LsT1')}_inv (KCSpu) /\LsT2':=xor (CSts1,xor (CSotp',CSpseudn'))
 /\LsT3':=xor (LsT2',LSpseudn) /\SigLS5':=xor ({ECDSA (Datas.LsT3')}_inv (KCSpu),SigLS4')
% LS sends (LSm) to CS
/\SND (CS.SigLS5'.SS.LSrn'.Datas) /\witness (LS,CS,auth7,{SigLS5,LSts4,LSrn'})
end role
```

Figure 12. *LS* server role in HLPSL.

As shown in Figure 12, the *LS* receives the start signal. Then, the *LS* generates and sends new *LSotpi* for all sensors (*SNi* and *CHi*). *LSotpi* includes new *SigLsEi* and pseudonym signature. Figures 10 and 11 both show that *SNi* and *CHi* receive *LSotp* from the *LS*. Furthermore, *SNi* and *CHi* use freshness nonces, timestamp, and signature to support reliable security. For instance, *SNi* uses *SigLsEi*, *SigSnEi*, *SNts*, and *SigSN* to achieve security processes with the *CHi* and *LS*. *SNi* collects data and uses one ECDSA signature with XoR operations to protect collected data and send it to the *CHi*. At this stage, the *CHi* aggregates data and adds security parameters. The *CHi* sends aggregation data to the *LS*. After that, the *LS* uses *LSotp*, *SigLS*5, and *LSrn* to connect with the *CS* securely. Figure 13 shows the *CS* with the storage process. The *CS* receives *LSotp* and uses ECDSA (*CsT*2), *CsT*1, *CsT*2, and *CSrn* to secure communication with the *LS*. Figure 14 shows session and environment roles as well as security goals (secrecy and authentication). REISCH applies seven secrecy and seven authentication goals. For instance, Sec1 represents secrets between *SNi* and the *LS* such as *SigSN*, *SigLsEi*, and *SigSnEi*. In addition, the authentication goal, such as auth4 proves freshness between the *CHi* and *LS* such as *CHts*2, *CHotp*, *CHrni*, and *SNrni*. Additionally, the environment role includes many attacks (replay, MITM, and impersonating) to test the security level in the REISCH scheme.

Results

The AVISPA tool describes the simulation results. We applied AVISPA with OFMC and CL-AtSe backends. The results of both of OFMC (Figure 15) and CL-AtSe (Figure 16) demonstrate to that the REISCH scheme is safe against passive and active attacks (as in the SUMMARY Section). Furthermore, Figures 15 and 16 show analysis details about simulation reports such as number of sessions, goals and statistical numbers. Moreover, the goals of authentication and secrecy in Figure 14 are applied to prevent the penetration of sensors' data/information in the network. These results prove that REISCH is reliable in combatting known attacks such as replay, MITM, and impersonating.

5.1.3. Security Comparison

In this section, we discuss the superiority of REISCH over existing schemes in terms of security (Table 3 shows a comparison of security features between our scheme and existing schemes). Compared with the scheme in [17] that uses a small key ($F_{2^{163}}$) and is extremely vulnerable to attacks, REISCH uses a key with 256 bit that resists attacks (reputable organization recommendations). REISCH also uses BLAKE2bp to get rid of hash attacks (collision and preimage) while the scheme in [18] focused on SHA1 performance without attention to the collision/preimage threats. In addition, all security parameters in REISCH such as SN_i's location are completely hidden, while the scheme in [19] transfers some information explicitly, such as ID (the elliptic curve parameters), in the registration and authentication phases.

```
role centralServer (CS,LS:agent, KCSpu,KLSpu:public_key, ECDSA:hash_func,SND,RCV:channel (dy))
played_by CS def=
local
     State:nat,
     SigCS1,SigCS2,LSts4:text, LSpseudo,LSpseudn,CSpseudo,CSpseudn:text,
     CsT1,CsT2,CsT3,CsT4,LSts2,CSts1,CSts2,CSotp,CSrn,LSrn,SS,Datas,LSotp,SigLSi,SigLS5:text
const
     sec6,sec7,auth5,auth6,auth7:protocol_id
init
     State := 0
transition
% CS receives from LS
1.State=0
  /\RCV (CS.LSotp'.SS)=|> State':=1 /\CSts1':=new ()/\CSpseudn':=new ()
  /\CSotp':=new ()/\CSrn':=new () /\CsT1':=xor (CSts1',xor (CSotp',xor (LSpseudn,CSrn')))
  /\CsT2':=xor (CSts1',xor (CSotp',xor (CSpseudn,LSpseudn)))
  /\request (CS,LS,auth5,{CSpseudo,LSpseudn,LSts2})
  /\secret ({LSpseudn,CSpseudn},sec6,{CS,LS}) /\secret ({SigLSi,CSotp'},sec7,{CS,LS})
% CS sends to LS
  /\SND (LS.{ECDSA (CsT2')}_inv (KLSpu).CsT1'.CsT2'.SS.CSrn')
  /\witness (CS,LS,auth6,{CSpseudn,LSpseudn,CSts1'})

% CS receives from LS
2.State=1
  /\RCV (CS.SigLS5'.SS'.LSrn'.Datas')=|> State':=2/\CSts2':=new ()
  /\SigCS1':={ECDSA (xor (LSts4,xor (LSpseudn,xor (LSrn,CSrn))))}_inv (KCSpu)
  /\CsT3':=xor (CSts1,xor (CSotp,CSpseudn)) /\CsT4':=xor (LSpseudn,CsT3')
  /\SigCS2':=xor ({ECDSA (Datas.CsT4')}_inv (KCSpu),SigCS1')
  /\request (CS,LS,auth7,{SigLS5,LSts4,LSrn'})
end role
```

Figure 13. *CS* server role in HLPSL.

```
role session (SNi,CHi,LS,CS:agent, KSNpu,KCHpu,KLSpu,KCSpu:public_key, ECDSA:hash_func ,MAXv:message,
                      SNpseudi,SNsli,Datai:text)
def=
local
     SND1,RCV1,SND2,RCV2,SND3,RCV3,SND4,RCV4:channel (dy)
composition
     sensori (SNi,CHi,LS,KSNpu,KCHpu,KLSpu,ECDSA,SNpseudi,SNsli,Datai,SND3,RCV3)
     /\clusterHeadi (CHi,LS,SNi,KCHpu,KLSpu,KSNpu,ECDSA,SNpseudi,SNsli,MAXv,SND2,RCV2)
     /\localServer (LS,CHi,SNi,CS,KLSpu,KCHpu,KSNpu,KCSpu,ECDSA,SNpseudi,SNsli,SND1,RCV1)
     /\centralServer (CS,LS,KCSpu,KLSpu,ECDSA,SND4,RCV4)
end role

role environment ()
def=
const
     kSNpu,kCHpu,kLS,kCS,ki:public_key, ecdsa:hash_func,datai,snpseudi,snsli:text,
     sni,chi,ls,cs,i:agent,maxV:message,
     sec1,sec2,sec3,sec4,sec5,sec6,sec7,auth1,auth2,auth3,auth4,auth5,auth6,auth7:protocol_id
     intruder_knowledge = {sni,chi,ls,i,kSNpu,kCHpu,kLS,ki}
composition
     session (sni,chi,ls,cs,kSNpu,kCHpu,kLS,kCS,ecdsa,maxV,datai,snpseudi,snsli)
     % Check replay attack
     %/\ session (sni,chi,ls,cs,kSNpu,kCHpu,kLS,kCS,ecdsa,maxV,datai,snpseudi,snsli)
     % Check MITM attack
     %/\ session (chi,sni,ls,cs,kSNpu,kCHpu,kLS,kCS,ecdsa,maxV,datai,snpseudi,snsli)
     % Check impersonate SNi
     /\ session (i,chi,ls,cs,kSNpu,kCHpu,kLS,kCS,ecdsa,maxV,datai,snpseudi,snsli)
     % Check impersonate CHi
     %/\ session (sni,i,ls,cs,kSNpu,kCHpu,kLS,kCS,ecdsa,maxV,datai,snpseudi,snsli)
     % Check impersonate LS
     %/\ session (sni,chi,i,cs,kSNpu,kCHpu,kLS,kCS,ecdsa,maxV,datai,snpseudi,snsli)
end role

goal
     secrecy_of sec1,sec2,sec3,sec4,sec5,sec6,sec7
     authentication_on auth1,auth2,auth3,auth4,auth5,auth6,auth7
end goal
environment ()
```

Figure 14. Supporting roles in HLPSL.

```
% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/span/span/testsuite/results/REIESCH_new11.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 532.61s
  visitedNodes: 1899 nodes
  depth: 7 plies
```

Figure 15. Simulation result using OFMC.

```
SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/span/span/testsuite/results/REIESCH_new11.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 325 states
  Reachable  : 325 states
  Translation: 0.12 seconds
  Computation: 0.01 seconds
```

Figure 16. Simulation result using CL-AtSe.

Table 3. Comparison of security features.

Security Feature	Fan and Gong [17]	Lavanya and Natarajan [19]	Staudemeyer et al. [20]	Malathy et al. [21]	Sharavanan et al. [22]	Sui and de Meer [23]	Hathaliya et al. [24]	Furtak et al. [25]	REISCH
Anti MITM		✓				✓	✓	✓	✓
Anti replay	✓	✓	✓			✓	✓	✓	✓
Availability	✓	✓					✓	✓	✓
Anti Sybil	✓							✓	✓
Anti Wormhole								✓	✓
Anti fake sink								✓	✓
Anti repository attack					✓				✓
Anti eavesdropping		✓	✓	✓	✓	✓	✓	✓	✓
Anti node replication								✓	✓
Anti collision/preimage		✓							✓
Pseudonym						✓	✓		✓
Homomorphic						✓			✓
Mutual authentication		✓					✓		✓

This allows intruders to distinguish a specific SN_i. Additionally, this scheme did not address the problem of hiding the SN_i location. Although the authors of [20] addressed privacy in their scheme's architecture to protect the SN_i parameters. Their scheme did not use the signatures camouflage or SN_{OTP} that are used in REISCH to support the privacy of data signing. This makes the privacy parameters in their scheme vulnerable to analysis and easy tracking. Furthermore, REISCH outperforms the scheme in [21], which did not use the signature aggregation scheme to support security and hide signatures. The scheme in [22] uses ECDSA to secure heterogeneous network environments. However, their scheme gives medical evaluators privileges to modify the medical parameters in the monitoring environment, SN_i's locations and even creates keys that could be the cause of an internal attack. Moreover, some information sent from SN_i to the server can clearly leak to intruders. Fortunately, REISCH does not suffer from these problems. REISCH adds sufficient randomization to hide security parameters, and patient records are protected even after LS is penetrated, while the scheme in [23] needs to support randomization and protect user information when a demand–response management unit is penetrated by an intruder. Besides, an intruder can send messages from a forged unit and deceive users after penetrating this module and revealing information. REISCH is robust against information leakage, while the scheme in [24] uses a 160-bit key that is vulnerable to attacks. It explicitly sends patient identities within the encrypted message in the login and authentication phases. If an intruder can break the encryption, he/she can use this information in data disclosure. REISCH uses ECDSA-BLAKE2bp and random pseudonyms to secure data signing. The scheme in [25] is based on SHA1 and HMAC, which are vulnerable to attacks in signing and authenticating collected data. It also does not include a pseudonym mechanism to protect SN_i parameters from misbehaving.

5.2. Performance Analysis

In this section, the theoretical and experimental performance analysis is presented to examine the computation processes of REISCH in improving the performance of the HWSN lifetime.

5.2.1. Theoretical Analysis

REISCH uses several features that qualify it to be efficient in HWSN performance. First, it relies on the ECDSA algorithm that integrates data collected by small keys compared to public key cryptography algorithms (RSA, DSA and Elgamal). For instance, ECDSA produces 256-bit equivalent keys in security for 3072-bit keys produced by RSA, DSA, and Elgamal. Second, REISCH implicitly uses BLAKE2bp with ECDSA, which is dramatically efficient in the operation of a hash function instead of SHA1. Third, REISCH uses the homomorphic property to combine signatures in CHs and significantly reduces energy dissipation. Fourth, REISCH relies on the LEACH routing protocol, which is the most efficient energy-saving protocol in WSN. Fifth, REISCH relies on rapid random pseudonyms to protect medical records rather than complex and costly processes of encryption and anonymity. Finally, REISCH uses XML to support efficient patient data management. Therefore, these features allow REISCH to maintain the energy of the SNs as long as possible.

5.2.2. Experimental Analysis

In this section, we evaluate the performance of REISCH in the execution of security operations in conjunction with the collected and saved data. As noted in previous sections, SNs require performance-efficient signatures to perform services for as long as possible in patients' monitoring and care. We provide tests on hash algorithms (SHA and BLAKE) and the signature algorithm (ECDSA). Additionally, we applied these algorithms to HWSN to analyze performance properties such as time, storage, and energy. Table 4 shows all the simulation parameters used in HWSN, while Table 5 shows computational operations in the REISCH scheme. All hash and signature algorithms were implemented by C language while WSN was designed in Octave under Ubuntu 16.04 LTS, processor Intel Core i5 2.6 GHz, OS type 64-bit, Memory 4 GiB, and disk 32.0 GB.

Table 4. REISCH's simulation parameters.

Parameters	Value
Area of WSN	1000 m × 1000 m
Number of *SN*s	200
Number of *CH*s	5%
Number of hops	2
Node type	Homogeneous
Node distribution	Random
LS location	(500, 500)
Dif	Maximum value (707.1068)
Initial energy	25 J
Size of packet	200 K, 400 K, 800 K and 1 M
Control packet size	50 B
Rounds	1000
Routing protocol	LEACH
Propagation energy	10 nJ/bit/m^2
Multi-hop propagation energy	0.0013 pJ/bit/m^4
Aggregation energy	5 nJ/bit/signal
Number of runs	100
Simulation time	300 s
Simulator	Octave

Table 5. REISCH's computational processes.

Process Type	Number of Process			Running Time	Storage	Energy
	SN	*CH*	*LS*			
SHA1 hash	1	1	Many	0.05529	160 bits	0.008464
BLAKE2bp hash	1	1	Many	0.040606	512 bits	0.006216
Keys generation	2	2	2	0.000859	256 bits	0.000132
Point multiplication	2	2	Many	0.000543	-	0.000083
ECDSA-SHA1 signature	1	1	-	0.072838	256 bits	0.011151
ECDSA-SHA1 verification	-	-	Many	0.073103	-	0.011191
ECDSA-BLAKE2bp signature	1	1	-	0.050046	256 bits	0.007662
ECDSA-BLAKE2bp verification	-	-	Many	0.052076	-	0.007972

The computation of energy in our scheme is based on the Micaz sensor specification. This process uses parameters such as current (0.0567), voltage (2.7), and time to extract both power and energy using $power = current \times voltage$ and $energy = time \times power$. We relied on real data provided by the City of Melbourne that is licensed under CC 4.0 [77]. These data were generated by sensors to monitor environmental parameters such as humidity, temperature, and light, as well as to include some information such as timestamp and ID. We divided these data into different sizes (200 K, 400 K, 800 K, and 1 M) and then converted them into an XML context. We used a large data size such as 1 M to test signature processes and security parameters in consuming sensor energy and thus the applicability of WSN. Furthermore, there are no communication channels between patients and *SN*s. To check performance, we implemented the SHA1-160, SHA2-256, BLAKE2s-256, BLAKE2b-512, BLAKE2sp-256, and BLAKE2bp-512 algorithms with 1 MB data size, as shown in Figure 17. Moreover, Figure 18 shows that ECDSA-BLAKE2bp gives the best execution time of ECDSA-SHA1. In addition, Figures 19, 20, and 21 show execution time (minimum, maximum, and average) for hash functions when using 200 K, 400 K, 800 K, and 1 M data. We also notice that BLAKE2bp has the best performance in terms of execution time in all figures. Additionally, Figures 22, 23, and 24 show the execution time (minimum, maximum, and average) for the ECDSA algorithms when using 200 K, 400 K, 800 K, and 1 M data. Thus, the amendment to the ECDSA algorithm is entirely appropriate for the use of security measures with the longest life of the *SN*s from the original algorithm.

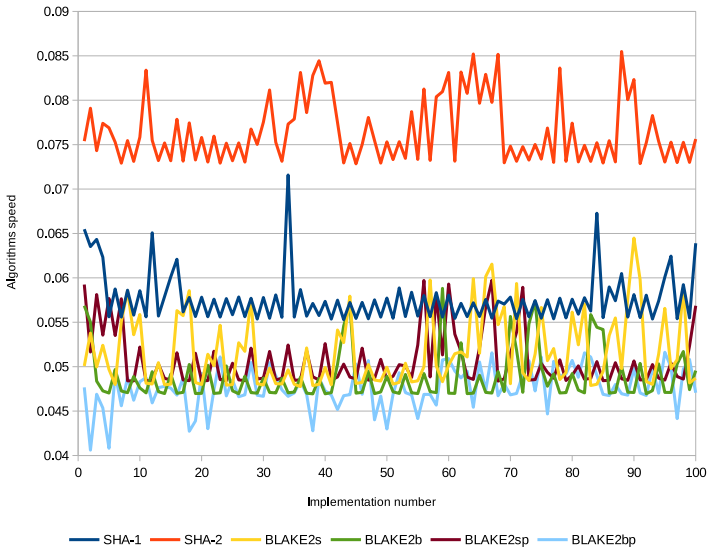

Figure 17. Comparison of SHA and BLAKE2 with 1 MB data.

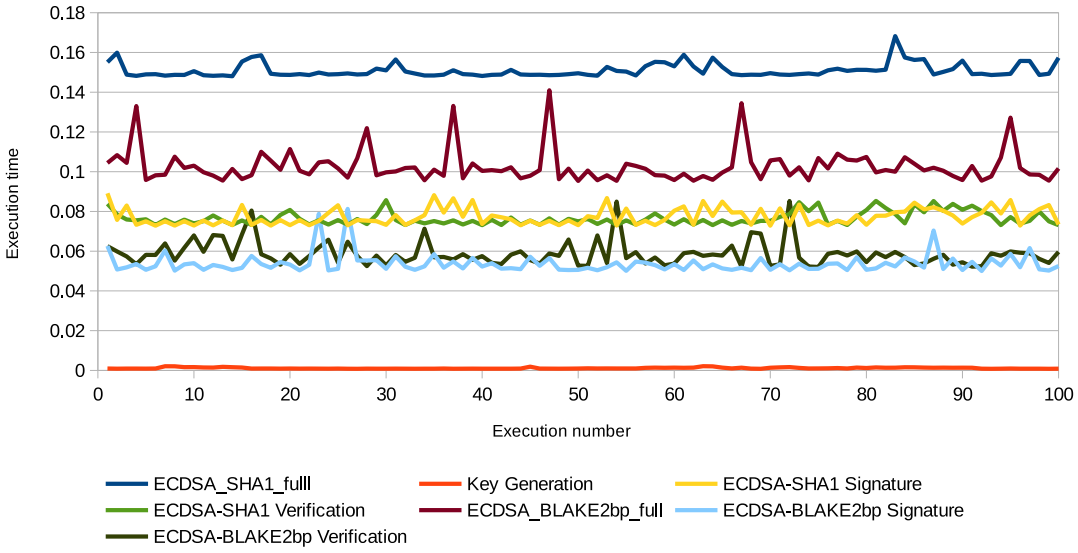

Figure 18. Execution time of ECDSA-SHA1 and ECDSA-BLAKE2bp with 1 MB data.

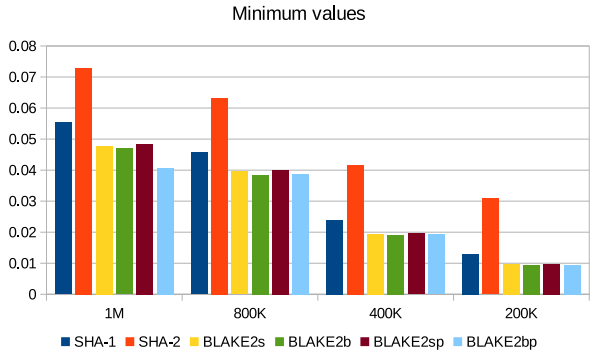

Figure 19. Minimum execution time of hash functions with different data sizes.

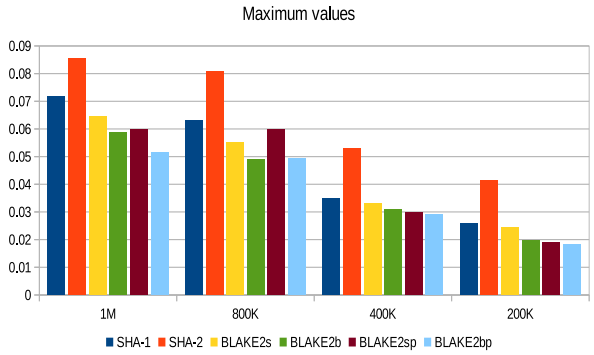

Figure 20. Maximum execution time of hash functions with different data sizes.

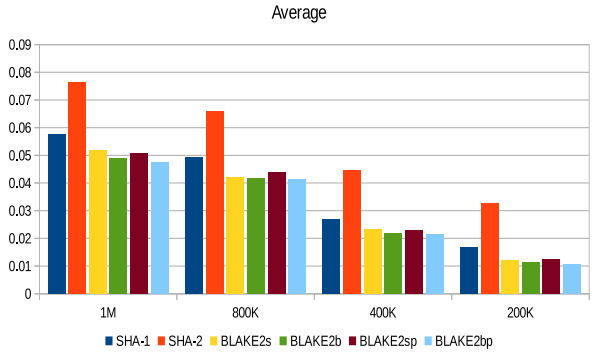

Figure 21. Average execution time of hash functions with different data sizes.

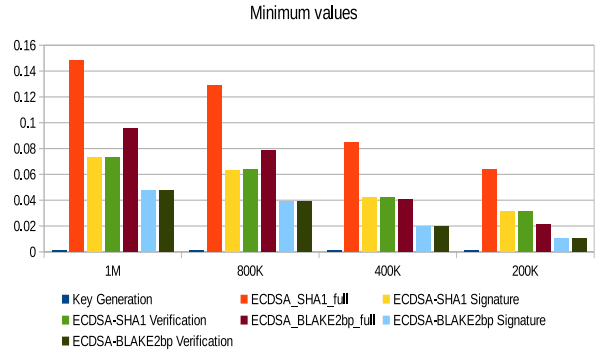

Figure 22. Minimum execution time of ECDSA algorithms with different data sizes.

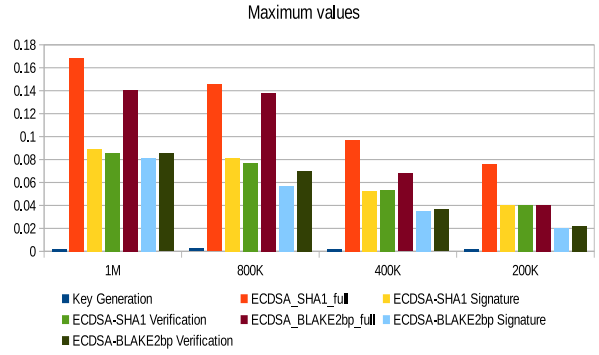

Figure 23. Maximum execution time of ECDSA algorithms with different data sizes.

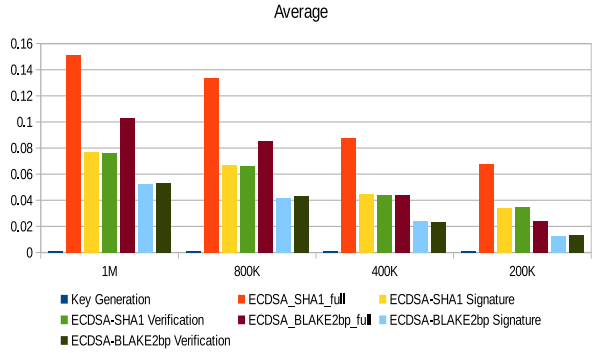

Figure 24. Average execution time of ECDSA algorithms with different data sizes.

We computed message complexity which is the number of messages transmitted between network entities. For each round, SN and CH send one message while CH and LS receive a set of aggregated messages. Thus, the message complexity with modified algorithms for SNs is (156,972), CHs is (8313), and LS is (165,285), while with original algorithms for SNs is (142,541), CHs is (7572) ,and LS is (150,113). Message overhead is to calculate the message size between network entities. In each round, the message overhead of SN is (1024 + 32) bytes while CH is (15,360 + 32) bytes. Figure 25 demonstrates that REISCH-ECDSA-BLAKE2bp is better than REISCH-ECDSA-SHA1 in terms of alive SNs, namely, HWSN will have a longer life span to collect patient data when it uses REISCH-ECDSA-BLAKE2bp. We noticed that REISCH with the modified algorithm (ECDSA-BLAKE2bp) has more alive SNs by 24% than the original algorithm (ECDSA-SHA1). Furthermore, the first SN dies when using the modified algorithm in round 322, while in the original algorithm is in round 295.

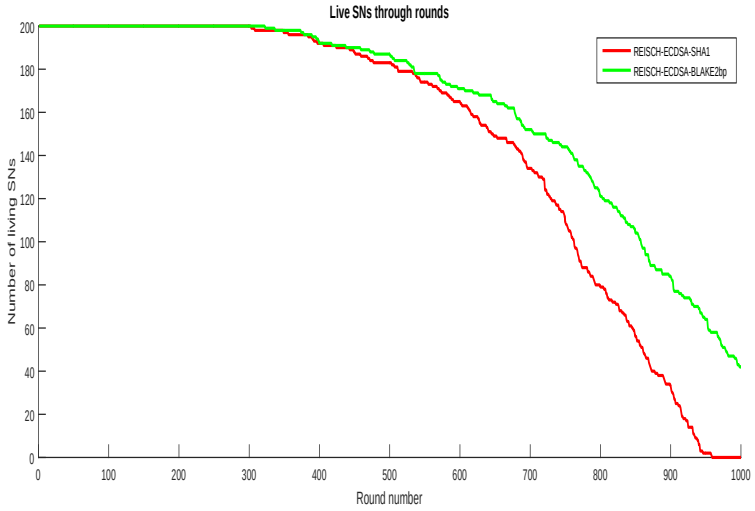

Figure 25. Comparison of alive SNs.

5.2.3. Performance Comparison

In this section, we discuss the superiority of REISCH to existing schemes in terms of performance (Table 6 shows a comparison of the ECDSA's signature and verification (running time) between our scheme and existing schemes). Due to different environments, security parameters and network parameters such as key length, number of SNs, etc., it is difficult to compare schemes' performances. However, we made some comparisons to illustrate the superiority of REISCH on the existing schemes in terms of performance. The scheme in [17] focused on accelerating ECDSA's verification based on computation results for neighboring SNs. These computations consume additional energy. In addition, this scheme is vastly expensive if applied to a cluster scheme because CH needs to accomplish one PM

in each signature for each SN_i and will thus consume energy in the intermediate SNs. REISCH does not need these computations because signatures' verification is performed in LS. Schemes in [18,19,21] used ECDSA to sign data without using homomorphic property. Consequently, the performance of the SNs would be remarkably low due to signature and verification processes in each round. The scheme in [18] addressed the bits (8 and 32) of data processing in SHA1 but did not address the cost of energy consumption by SHA1. In addition, the scheme in [19] did not support the clustering environment to reduce energy consumption and the computation time to generate and verify the signature which was not clearly indicated. Furthermore, the scheme in [22] used SHA2, which is more secure than SHA1 but performs heavy processes that significantly affect the energy of SNs. It also addresses only computations in transport while REISCH addresses computations in transport and processing using BLAKE2bp and homomorphic property. The schemes in [20,24,25] rely on the use of encryption to protect data without a homomorphic property, since encryption processes extremely consume SNs resources (as mentioned in Section 3.3, Point 3), while REISCH uses signatures and homomorphic property to improve HWSN network performance. Although the scheme in [23] uses encryption and signature of data with homomorphic properties, encryption can particularly affect network performance, especially through a burden on the servers. Moreover, the scheme in [25] has implemented the RSA-2048 bits algorithm, which is significantly expensive in encryption operations. In addition, it uses several parameters such as many keys, 2048-bit key length, and SN_i addresses (master, replica, and gateway) that cause storage problems in the pre-deployment and registration phases (consumption of SNs resources). It uses a random routing of the sensor network without relying on a specific routing protocol such as LEACH. This scheme considers the structure of the data in the SN_i memory and does not pay attention to the structure of the data as they are transferred to the servers. REISCH uses XML to support performance of the LS and CS without having to convert data formats between network devices. In terms of alive SNs, REISCH provides more than 24% while the method in [78] 17.5%, the method in [79] 18.26% (100 nodes), the method in [80] 16% (100–700 nodes), and the method in [81] 7.14% (100 nodes) and 4% (50 nodes). Thus, REISCH provides longer network lifetime than the schemes in [78–81]. Recent research (e.g., [26–30]) has used different ways to improve ECDSA's procedures. However, REISCH provides better performance in terms of ECDSA's signature and verification than existing schemes (as shown in Table 6).

Table 6. Comparison of ECDSA's procedures.

Running Time (s)	Fan and Gong [17]	Kodali [18]	Malathy et al. [21]	Kittur and Pais [26]	Kuang et al. [27]	Marino et al. [28]	Zhao et al. [29]	Liu et al. [30]	REISCH
Signature	0.38	0.941	0.59	0.078	0.3472	0.434	0.084	0.051	0.050
Verification	0.65	-	-	0.079	-	0.429	0.088	0.105	0.052

6. Conclusions and Future Work

Wireless sensor networks provide unique and important care services when used with EMRs. Unfortunately, these networks suffer from performance and security problems, as mentioned in the previous sections. Therefore, we propose a REISCH scheme to address performance and security problems and cover gaps in existing research. As a result, REISCH uses ECDSA-BLAKE2bp and provides the best performance from using the original ECDSA-SHA1 algorithm. REISCH with the modified algorithm saves more than 24% alive SNs. In addition, the results of the security analysis prove that REISCH is safe against attacks in the threat model. Future directions planned for the development of this scheme are as follows:

1. Our scheme requires security mechanisms to support authentication requests (such as encryption and mutual authentication) and authorization (access control models) and thus allow legitimate users (patients and providers) to access medical records on remote servers (AS and DS).
2. Support for our scheme is by using ECDSA-BLAKE2bp with efficient curves such as the Edward curve and efficient PM methods such as Frobenius to improve the efficiency of patients' data signing in HWSN.
3. We intend to integrate our scheme into a real HWSN environment to evaluate the efficiency and feasibility of REISCH algorithms to improve the lifetime of SNs in patients' data collection as long as possible.

Author Contributions: Conceptualization, M.A. and Z.Z.; methodology, M.A.; software, M.A.; formal analysis, Z.Z.; writing—original draft preparation, M.A.; writing—review and editing, M.A., Z.Z., and J.Z.; supervision, Z.Z. and J.Z.; and project administration, M.A.

Funding: This research received no external funding.

Acknowledgments: We would like to acknowledge and thank the efforts of Barbara Harmes who revised our paper as well as the valuable feedback of the reviewers.

Conflicts of Interest: The authors declare that they have no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

SN, CH	Sensor, Cluster Head
LS, CS	Local Server, Central Server
K_{pu_i}, K_{pr_i}	Public and private keys
OTP	One time passcode
$Pseud$	Pseudonym generated by entities (SN, CH, LS, CS)
$Parity$	The value specifies the signature of even/odd
P	Entity parameters
RN	The random number generated by entities
TS	Timestamp generated by entities
$SigSN, SigCH$	Signatures generated by SN, CH
$SigLS, SigCS$	Signatures generated by LS, CS
$SigSnEi, SigLsEi$	Random ephemeral value the same length as the signature generated by SN, LS
$S_N C_H D$	Distance between SN and CH
$S_N L_S D$	Distance between SN and LS
Dif	Value proves SN in the HWSN's area
SL	Sensor location
m	Message sent by entity
A	Aggregation function
$h(.)$	One-way hash function
$\|, \oplus$	Concatenation and exclusive or operations

References

1. Sarkar, B.K. Big data for secure healthcare system: a conceptual design. *Complex Intell. Syst.* **2017**, *3*, 133–151.
2. Kumar, P.; Lee, H.-J. Security issues in healthcare applications using wireless medical sensor networks: A survey. *Sensors* **2011**, *12*, 55–91.
3. Al Ameen, M.; Liu, J.; Kwak, K. Security and privacy issues in wireless sensor networks for healthcare applications. *J. Med. Syst.* **2012**, *36*, 93–101.
4. Ayyildiz, C.; Erdem, H.E.; Dirikgil, T.; Dugenci, O.; Kocak, T.; Altun, F.; Gungor, V.C. Structure health monitoring using wireless sensor networks on structural elements. *Ad Hoc Netw.* **2019**, *82*, 68–76.
5. Javadi, S.S.; Razzaque, M. Security and privacy in wireless body area networks for health care applications. In *Wireless Networks and Security*; Springer: Berlin, Germany, 2013; pp. 165–187.
6. Manogaran, G.; Varatharajan, R.; Lopez, D.; Kumar, P.M.; Sundarasekar, R.; Thota, C. A new architecture of Internet of Things and big data ecosystem for secured smart healthcare monitoring and alerting system. *Future Gener. Comput. Syst.* **2018**, *82*, 375–387.
7. Bruland, P.; Doods, J.; Brix, T.; Dugas, M.; Storck, M. Connecting healthcare and clinical research: Workflow optimizations through seamless integration of EHR, pseudonymization services and EDC systems. *Int. J. Med. Inf.* **2018**, *119*, 103–108.
8. Chuang, M.-C.; Chen, M.C. An anonymous multi-server authenticated key agreement scheme based on trust computing using smart cards and biometrics. *Expert Syst. Appl.* **2014**, *41*, 1411–1418.
9. Griggs, K.N.; Ossipova, O.; Kohlios, C.P.; Baccarini, A.N.; Howson, E.A.; Hayajneh, T. Healthcare blockchain system using smart contracts for secure automated remote patient monitoring. *J. Med. Syst.* **2018**, *42*, 130.
10. Al-Turjman, F.; Alturjman, S. Confidential smart-sensing framework in the IoT era. *J. Supercomput.* **2018**, *74*, 5187–5198.
11. Verma, G.K.; Singh, B.; Singh, H. Bandwidth efficient designated verifier proxy signature scheme for healthcare wireless sensor networks. *Ad Hoc Netw.* **2018**, *81*, 100–108.
12. Aceto, G.; Persico, V.; Pescapé, A. The role of information and communication technologies in healthcare: Taxonomies, perspectives, and challenges. *J. Netw. Comput. Appl.* **2018**, *107*, 125–154.
13. Gao, Y.; Ao, H.; Feng, Z.; Zhou, W.; Hu, S.; Tang, W. Mobile Network Security and Privacy in WSN. *Proc. Comput. Sci.* **2018**, *129*, 324–330.
14. Li, J.; Zhang, W.; Kumari, S.; Choo, K.K.R.; Hogrefe, D. Security analysis and improvement of a mutual authentication and key agreement solution for wireless sensor networks using chaotic maps. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, e3295.
15. Al-Zubaidie, M.; Zhang, Z.; Zhang, J. PAX: Using Pseudonymization and Anonymization to Protect Patients' Identities and Data in the Healthcare System. *Int. J. Environ. Res. Public Health* **2019**, *16*, 1–36.
16. Pawar, P.M.; Nielsen, R.H.; Prasad, N.R.; Prasad, R. GSHMAC: Green and Secure Hybrid Medium Access Control for Wireless Sensor Network. *Wirel. Pers. Commun.* **2018**, *100*, 267–281.
17. Fan, X.; Gong, G. Accelerating signature-based broadcast authentication for wireless sensor networks. *Ad Hoc Netw.* **2012**, *10*, 723–736.
18. Kodali, R.K. Implementation of ECDSA in WSN. In Proceedings of the 2013 International Conference on IEEE Control Communication and Computing (ICCC), Thiruvananthapuram, India, 13–15 December 2013; pp. 310–314.
19. Lavanya, M.; Natarajan, V. LWDSA: Lightweight digital signature algorithm for wireless sensor networks. In Sādhanā; Springer: Berlin, Germany, 2017; pp. 1–15.
20. Staudemeyer, R.C.; Pöhls, H.C.; Wójcik, M. The road to privacy in IoT: beyond encryption and signatures, towards unobservable communication. In Proceedings of the 2018 IEEE 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), Chania, Greece, 12–15 June 2018; pp. 14–20.
21. Malathy, S.; Geetha, J.; Suresh, A.; Priya, S. Implementing Elliptic Curve Cryptography with ACO Based Algorithm in Clustered WSN for Border Surveillance. In Proceedings of the IEEE 2018 Fourth International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB), Chennai, India, 27–28 February 2018; pp. 1–5.
22. Sharavanan, P.; Sridharan, D.; Kumar, R. A Privacy Preservation Secure Cross Layer Protocol Design for IoT Based Wireless Body Area Networks Using ECDSA Framework. *J. Med. Syst.* **2018**, *42*, 196.

23. Sui, Z.; de Meer, H. Bap: A batch and auditable privacy preservation scheme for demand-response in smart grids. *IEEE Trans. Ind. Inf.* **2019**, *16*, 842–853.
24. Hathaliya, J.J.; Tanwar, S.; Tyagi, S.; Kumar, N. Securing electronics healthcare records in Healthcare 4.0: A biometric-based approach. *Comput. Electr. Eng.* **2019**, *76*, 398–410.
25. Furtak, J.; Zieliński, Z.; Chudzikiewicz, J. A Framework for Constructing a Secure Domain of Sensor Nodes. *Sensors* **2019**, *19*, 2797.
26. Kittur, A.S.; Pais, A.R. A new batch verification scheme for ECDSA * signatures. *Sādhanā* **2019**, *44*, 157.
27. Kuang, B.; Fu, A.; Yu, S.; Yang, G.; Su, M.; Zhang, Y. Esdra: An efficient and secure distributed remote attestation scheme for IoT swarms. *IEEE Internet Things J.* **2019**, *6*, 8372–8383.
28. Marino, F.; Moiso, C.; Petracca, M. PKIoT: A public key infrastructure for the internet of things. *Trans. Emerg. Telecommun. Technol.* **2019**, *30*, e3681.
29. Zhao, Y.; Yu, Y.; Li, Y.; Han, G.; Du, X. Machine learning based privacy-preserving fair data trading in big data market. *Inf. Sci.* **2019**, *478*, 449–460.
30. Liu, Y.; Zhao, Y.; Tian, A.; Yu, Y.; Du, X. Blockchain based privacy-preserving software updates with proof-of-delivery for internet of things. *J. Parallel Distrib. Comput.* **2019**, *132*, 141–149.
31. Chiriaco, V.; Franzen, A.; Thayil, R.; Zhang, X. Finding partial hash collisions by brute force parallel programming. In Proceedings of the 2017 IEEE Long Island Systems, Applications and Technology Conference (LISAT), Farmingdale, NY, USA, 5 May 2017; pp. 1–6.
32. Merrill, N. Better Not to Know? The SHA1 Collision & the Limits of Polemic Computation. In Proceedings of the ACM 2017 Workshop on Computing Within Limits, Berkeley, California, USA, 22–24 June 2017; pp. 37–42.
33. Yang, Y.; Zhang, X.; Yu, J.; Zhang, P.; Chen, F. Research on the hash function structures and its application. *Wirel. Pers. Commun.* **2017**, *94*, 2969–2985.
34. Giechaskiel, I.; Cremers, C.; Rasmussen, K.B. When the Crypto in Cryptocurrencies Breaks: Bitcoin Security under Broken Primitives. *IEEE Secur. Priv.* **2018**, *16*, 46–56.
35. Park, S.y.; Kim, K. A study on the processing and reinforcement of message digest through two-dimensional array masking. In Proceedings of the IEEE 2018 International Conference on Information Networking (ICOIN), Chiang Mai, Thailand, 10–12 January 2018; pp. 540–544.
36. Beglaryan, M.; Petrosyan, V.; Bunker, E. Development of a tripolar model of technology acceptance: hospital-based physicians' perspective on EHR. *Int. J. Med. Inf.* **2017**, *102*, 50–61.
37. Alkureishi, M.A.; Lee, W.W.; Lyons, M.; Wroblewski, K.; Farnan, J.M.; Arora, V.M. Electronic-clinical evaluation exercise (e-CEX): a new patient-centered EHR use tool. *Pat. Educ. Couns.* **2018**, *101*, 481–489.
38. Senteio, C.; Veinot, T.; Adler-Milstein, J.; Richardson, C. Physicians' perceptions of the impact of the EHR on the collection and retrieval of psychosocial information in outpatient diabetes care. *Int. J. Med. Inf.* **2018**, *113*, 9–16.
39. Muthee, V.; Bochner, A.F.; Osterman, A.; Liku, N.; Akhwale, W.; Kwach, J.; Prachi, M.; Wamicwe, J.; Odhiambo, J.; Onyango, F.; et al. The impact of routine data quality assessments on electronic medical record data quality in Kenya. *PLoS ONE* **2018**, *13*, e0195362.
40. Heart, T.; Ben-Assuli, O.; Shabtai, I. A review of PHR, EMR and EHR integration: A more personalized healthcare and public health policy. *Health Policy Technol.* **2017**, *6*, 20–25.
41. Al-Zubaidie, M.; Zhang, Z.; Zhang, J. Efficient and Secure ECDSA Algorithm and its Applications: A Survey. *Int. J. Commun. Netw. Inf. Secur.* **2019**, *11*, 7–35.
42. Dou, Y.; Weng, J.; Ma, C.; Wei, F. Secure and efficient ECC speeding up algorithms for wireless sensor networks. *Soft Comput.* **2017**, *21*, 5665–5673.
43. Bachiller, Y.; Busch, P.; Kavakli, M.; Hamey, L. Survey: Big Data Application in Biomedical Research. In Proceedings of the ACM 2018 10th International Conference on Computer and Automation Engineering, Brisbane, Australia, 24–26 February 2018; pp. 174–178.
44. Hoceini, O.; Afifi, H.; Aoudjit, R. Authentication Based Elliptic Curves Digital Signature for ZigBee Networks. In *International Conference on Mobile, Secure, and Programmable Networking*; Springer: Berlin, Germany, 2017; pp. 63–73.
45. Abueh, Y.J.; Liu, H. Message authentication in driverless cars. In Proceedings of the 2016 IEEE Symposium on Technologies for Homeland Security (HST), Waltham, MA, USA, 10–11 May 2016; pp. 1–6.

46. Franeková, M.; Holečko, P.; Bubeníková, E.; Kanáliková, A. Transport scenarios analysis within C2C communications focusing on security aspects. In Proceedings of the 2017 IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI), Herl'any, Slovakia, 26–28 January 2017; pp. 000461–000466.
47. Shi, Z.; Ma, C.; Cote, J.; Wang, B. Hardware implementation of hash functions. In *Introduction to Hardware Security and Trust*; Springer: Berlin, Germany, 2012; pp. 27–50.
48. Luo, P.; Li, C.; Fei, Y. Concurrent error detection for reliable SHA-3 design. In Proceedings of the IEEE 2016 International Great Lakes Symposium on VLSI, Boston, MA, USA, 18–20 May 2016; pp. 39–44.
49. Luo, P.; Athanasiou, K.; Fei, Y.; Wahl, T. Algebraic fault analysis of SHA-3. In Proceedings of the IEEE 2017 Design, Automation & Test in Europe Conference & Exhibition (DATE), Lausanne, Switzerland, 27–31 March 2017; pp. 151–156.
50. Chaves, R.; Sousa, L.; Sklavos, N.; Fournaris, A.P.; Kalogeridou, G.; Kitsos, P.; Sheikh, F. Secure hashing: SHA-1, SHA-2, and SHA-3. In *Circuits and Systems for Security and Privacy*, Taylor & Francis Group: Abingdon, UK, 2016; pp. 105–132.
51. Dobraunig, C.; Eichlseder, M.; Mendel, F. Analysis of SHA-512/224 and SHA-512/256. In *International Conference on the Theory and Application of Cryptology and Information Security*; Springer: Berlin, Germany, 2015; pp. 612–630.
52. Al Maashri, A.; Pathuri, L.; Awadalla, M.; Ahmad, A.; Ould-Khaoua, M. Optimized hardware crypto engines for XTEA and SHA-512 for wireless sensor nodes. *Ind. J. Sci. Technol.* **2016**, *9*, 2016.
53. Lu, Y.; Zhai, J.; Zhu, R.; Qin, J. Study of wireless authentication center with mixed encryption in WSN. *J. Sens.* **2016**, *2016*, 1–7.
54. Saha, S.; Das, R.; Datta, S.; Neogy, S.; A cloud security framework for a data centric WSN application. In Proceedings of the ACM 17th International Conference on Distributed Computing and Networking, Singapore, 4 January 2016; pp. 1–6.
55. Aumasson, J.-P.; Henzen, L.; Meier, W.; Phan, R.C.-W. SHA-3 proposal blake. *NIST* **2008**, *229*, 1–48.
56. Cho, H. ASIC-resistance of multi-hash proof-of-work mechanisms for blockchain consensus protocols. *IEEE Access* **2018**, *6*, 66210–66222.
57. Aumasson, J.P.; Neves, S.; Wilcox-O'Hearn, Z.; Winnerlein, C. BLAKE2: Simpler, smaller, fast as MD5. In *International Conference on Applied Cryptography and Network Security*; Springer: Berlin, Germany, 2013; pp. 119–135.
58. Körber, O.; Keller, J.; Holmbacka, S. Energy-efficient Execution of Cryptographic Hash Functions on big. LITTLE Architecture. In Proceedings of the IEEE 2018 13th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), Lille, France, 9–11 July 2018; pp. 1–7.
59. Mozaffari-Kermani, M.; Azarderakhsh, R.; Aghaie, A. Fault detection architectures for post-quantum cryptographic stateless hash-based secure signatures benchmarked on ASIC. *ACM Trans. Embed. Comput. Syst. (TECS)* **2017**, *16*, 1–19.
60. Yang, Y.; Chen, F.; Sun, Z.; Wang, S.; Li, J.; Chen, J.; Ming, Z. Secure and efficient parallel hash function construction and its application on cloud audit. In *Soft Computing*; Springer: Berlin, Germany, 2018; pp. 1–19.
61. Neubauer, T.; Heurix, J. A methodology for the pseudonymization of medical data. *Int. J. Med. Inf.* **2011**, *80*, 190–204.
62. Zhou, J.; Cao, Z.; Dong, X.; Vasilakos, A.V. Security and privacy for cloud-based IoT: Challenges. *IEEE Commun. Mag.* **2017**, *55*, 26–33.
63. Vatsalan, D.; Sehili, Z.; Christen, P.; Rahm, E. Privacy-preserving record linkage for big data: Current approaches and research challenges. In *Handbook of Big Data Technologies*; Springer: Berlin, Germany, 2017; pp. 851–895.
64. Bogos, S.; Gaspoz, J.; Vaudenay, S. Cryptanalysis of a homomorphic encryption scheme. *Cryptogr. Commun.* **2018**, *10*, 27–39.
65. Chen, C.M.; Fang, W.; Wang, K.H.; Wu, T.Y. Comments on "an improved secure and efficient password and chaos-based two-party key agreement protocol". *Nonlinear Dyn.* **2017**, *87*, 2073–2075.
66. Jo, S.M.; Chung, K.Y. Design of access control system for telemedicine secure XML documents. *Multimed. Tools Appl.* **2015**, *74*, 2257–2271.
67. Emmanuel, N.; Khan, A.; Alam, M.; Khan, T.; Khan, M.K. Structures and data preserving homomorphic signatures. *J. Netw. Comput. Appl.* **2018**, *102*, 58–70.

68. Luo, F.; Wang, F.; Wang, K.; Chen, K. A more efficient leveled strongly-unforgeable fully homomorphic signature scheme. *Inf. Sci.* **2019**, *480*, 70–89.
69. Kapusta, K.; Memmi, G.; Noura, H. Additively homomorphic encryption and fragmentation scheme for data aggregation inside unattended wireless sensor networks. In *Annals of Telecommunications*; Springer: Berlin, Germany, 2019; pp. 1–9.
70. Awaad, M.H; Jebbar, W.A. Extending the WSN lifetime by dividing the network area into a specific zones. *Int. J. Comput. Netw. Inf. Secur.* **2015**, *7*, 33–39.
71. Awaad, M.H; Jebbar, W.A. Study to analyze and compare the leach protocol with three methods to improve it and determine the best choice. *J. Comput. Sci. Control Syst.* **2014**, *7*, 5–12.
72. Al-Zubaidie, M.; Zhang, Z.; Zhang, J. RAMHU: A New Robust Lightweight Scheme for Mutual Users Authentication in Healthcare Applications. *Secur. Commun. Netw.* **2019**, *2019*, 1–26.
73. Kumar, N.; Kaur, K.; Misra, S.C.; Iqbal, R. An intelligent RFID-enabled authentication scheme for healthcare applications in vehicular mobile cloud. *Peer-to-Peer Netw. Appl.* **2016**, *9*, 824–840.
74. Team, T.A. AVISPA v1.1 User Manual. Available online: http://www.avispa-project.org (accessed on 25 June 2019).
75. Iqbal, U.; Shafi, S. A Provable and Secure Key Exchange Protocol Based on the Elliptical Curve Diffe–Hellman for WSN. In *Advances in Big Data and Cloud Computing*; Springer: Berlin, Germany, 2019; pp. 363–372.
76. Ostad-Sharif, A.; Arshad, H.; Nikooghadam, M.; Abbasinezhad-Mood, D. Three party secure data transmission in IoT networks through design of a lightweight authenticated key agreement scheme. *Futur. Gener. Comput. Syst.* **2019**, *100*, 882–892.
77. City of Melbourne Open Data Team. Sensor Readings, with Temperature, Light, Humidity every 5 Minutes at 8 Locations. 19 October 2018. Available online: https://data.melbourne.vic.gov.au/Environment/Sensor-readings-with-temperature-light-humidity-ev/ez6b-syvw (accessed on 18 May 2019).
78. Elhoseny, M.; X.; El-Minir, H.K.; Riad, A.M. An energy efficient encryption method for secure dynamic WSN. *Secur. Commun. Netw.* **2016**, *9*, 2024–2031.
79. Elhoseny, M.; Elminir, H.; Riad, A.; Yuan, X. A secure data routing schema for WSN using elliptic curve cryptography and homomorphic encryption. *J. King Saud Univ. Comput. Inf. Sci.* **2016**, *28*, 262–275.
80. Prithi, S.; Sumathi, S. LD2FA-PSO: A novel learning dynamic deterministic finite automata with pso algorithm for secured energy efficient routing in wireless sensor network. *Ad Hoc Netw.* **2020**, *97*, 102024.
81. Vinitha, A.; Rukmini, M.S.S. Secure and energy aware multi-hop routing protocol in WSN using taylor-based hybrid optimization algorithm. *J. King Saud Univ. Comput. Inf. Sci.* 2019, 1–12.