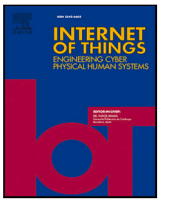
Internet of Things 25 (2024) 101041

Contents lists available at ScienceDirect

Internet of Things

journal homepage: www.elsevier.com/locate/iot



Research article

Lightweight encryption with data and device integrity using NLFSR and PUF for the Internet of Medical Things

Punam Kumari, Bhaskar Mondal *

Department of Computer Science and Engineering, National Institute of Technology Patna, Patna, 800005, Bihar, India

ARTICLE INFO

MSC:
0000
1111

Keywords:
Authentication
NLFSR
PUF
Pseudorandom
Stream cipher

ABSTRACT

Nowadays the Internet of Medical Things (IoMT) is in top demand; they communicate with each other in an IoT network, but there are various security issues including privacy preservation, data and device integrity, and authentication. The IoMT devices have challenges like limited computational power and resources, so conventional techniques are not suitable for them. This paper presents a lightweight stream cipher to ensure privacy protection, data integrity, and device integrity using a message authentication code (MAC). The stream cipher is designed using a nonlinear feedback shift register (NLFSR). The cipher is associated with a MAC generated by concatenating a physical unclonable function (PUF) response and the hash of the plaintext to protect the device and data integrity. The use of NLFSR and PUF makes the proposed scheme robust and lightweight for resource-constrained devices like MIoT. To evaluate the quality of the cipher the NIST 800-22 statistical test suite (NIST STS) is used, and results are compared with existing stream ciphers like Rivest cipher 4 (RC4), RC4 based on chaotic mapping (RCCM), and product algebra RCCM (PARCCM). All the tests and comparisons were performed using a Heart Disease Dataset. The scheme ensures the confidentiality and integrity of sensitive medical data which are critical for protecting patient privacy and regulatory compliance. The proposed scheme can also resist cloning attacks on the IoMT devices.

1. Introduction

Internet of Medical Things (IoMT) is an application of the IoT for medical and health-related data collection, monitoring systems, and wireless body area networks [1]. The IoMT has been referenced as smart healthcare, as technology for creating a digitalized healthcare system, connecting available medical resources and healthcare services.

Introducing Artificial Intelligence into the IoMT opens many opportunities to provide real-time, remote measurement and analysis of patient data by paving the way for other technologies and better access to healthcare through connectivity. A few important IoMT applications are diagnostics, recuperation, chronic care, and prophylactics. The key benefits of IoMT are that it benefits from real-time interventions in emergency circumstances, decreases costs, and reduces morbidity and financial burden due to fewer follow-up visits. It also supports healthcare service providers by optimizing resource and infrastructure use and reducing response time in the event of a medical emergency.

However, IoMT devices are frequently put in open and public areas, exposing them to cloning and physical attacks. Consequently, any security solution developed for IoMT devices must be efficient and detect any breaches in the physical security of IoT devices. So, authentication is especially important for edge devices [2]. Furthermore, thousands of IoT devices will likely be implemented in the coming years; these limited-resource devices are frequently coupled to sensors and actuators in process control plants. The

* Corresponding author.
E-mail address: bhaskar.cs@nitp.ac.in (B. Mondal).

https://doi.org/10.1016/j.iot.2023.101041
Received 12 July 2023; Received in revised form 18 November 2023; Accepted 21 December 2023
Available online 27 December 2023

problem with having such limited features, and low profile, is that many encryption algorithms, particularly public cryptography algorithms, are not suitable for them.

The traditional techniques store a secret key in a Flash or EEPROM memory present in the device and then use that secret key with lightweight ciphers that have been devised. Physically Unclonable Functions (PUFs) [3] are a promising hardware primitive used for IoT security without keeping any key stored in EEPROMs, or other expensive hardware [4]. Motivated by the earlier reported works, this paper presents a secure lightweight encryption scheme based on NLFSR-based stream cipher for IoMT devices and protects them from cloning and physical attack by using Arbiter PUF.

A newly designed NLFSR-based stream cipher encryption method is proposed in this paper for IoMT devices. The proposed scheme uses Arbiter PUF and a lightweight hash function (Neeva hash function) to verify data and device integrity. The following contributions to the paper are:

- Design an NLSFR-based stream cipher to encrypt IoMT data ensuring privacy protection.
- Design a MAC mechanism using Arbiter PUF (APUF). The APUF generates a parameter for the MAC.
- The MAC values are computed using a Lightweight hash function (Neeva hash function) algorithm and used to assure data and device integrity for secure communication.

The remaining parts of the work are completed in the following sections. The preliminaries of NLFSR, Neeva lightweight hash function, and Arbiter PUF are discussed in Section 3. The proposed scheme is presented in Section 4. Section 5 presents the results of the security analysis test performed on the proposed scheme. Lastly, the conclusion is presented in 6.

2. Related work

Hu et al. [5] have developed a Prefix Verifiable message authentication code (MAC) to secure data privacy and integrity in Location-based Services (LBS). LBS is a software service used to provide related location and information services to users. There may be possible location privacy attacks based on the location-sharing mechanism. Li et al. [6] have proposed a cumulative MAC scheme for the large MAC output. It is divided into two distinct parts: aggregation and accumulation.

Liu et al. [7] have designed a new stream cipher with the help of combined Rivest Cipher 4 RC4 and logistic map stream cipher named RC4 based on Chaotic Mapping (RCCM). They have generated another stream cipher by combining product algebra with a logistic map. The name of the newly developed stream cipher is Product Algebra RCCM (PARCCM). Applying this cipher improves the randomness properties of the RCCM-generated key sequences. These two ciphers have been used to encrypt the real dataset. After that, the NIST STS was applied to the keystream bit generated from these ciphers.

Zeng et al. [8] have proposed a Reconfigurable feedback shift register (RFSR) based on Grain cipher. These designs use a 32 bit Linear feedback shift register (LFSR) and a 64 bit NLFSR, whereas Grain-128 uses a 128 bit LFSR and NLFSR. The sensor node requires multiple RFSR ciphers to data link layer pairwise encryption and application layer encryption. The presented implementation is based on a basic cipher infrastructure, and the structure loads specific cipher data from Random-access memory as needed.

Parah et al. [9] developed an image encryption scheme that uses left data mapping (LDM), pixel repetition technique (PRT), RC4 encryption, and checksum computation. The input image of size $m \times n$ is enlarged using PRT, and secret binary data is encrypted using the RC4 encryption algorithm before being grouped into 3 bit pieces and transferred to decimal equivalents. The cover image is divided into 2×2 blocks to embed the shifted data, and two digits are embedded into the counter diagonal pixels in each block. A checksum digit computed from the block is embedded into one of the main diagonal pixels to identify and localize tampering. A fragile logo has been inserted into the cover medium, in addition to electronic health records, to enable data authentication at the receiver end.

Wu et al. [2] have provided attribute-based privacy preservation based on blockchain-enabled dynamic access control architecture integrated with Local Differential Privacy (LDP) techniques. LDP has an additional constraint that even if an attacker can access an individual's responses in the database, the adversary cannot learn too much about the user's data. Yanambaka et al. [10,11] proposed a PUF-based authentication protocol for IoMT, employing a secure database as a third entity for storing CRPs of IoMT devices. In these schemes, both the IoMT and the server were outfitted with a PUF circuit.

Mustapa et al. [12] have introduced a hardware authentication scheme using Ring oscillator PUFs (ROPUFs) for advanced meter infrastructure (AMI). In these schemes, the Utility Company (UC) works like a center point because it monitors and updates the customers regularly. All data is transferred to the UCs via the number of Smart Meters (SMs) and Data Collectors (DCs). In this network, each device is connected to ROPUF chips except UC. Thus, UC keeps track of the devices in the AMI. When data must be sent to the smart meter, it must authenticate itself—firstly, the smart meter requests to UC for secure communication establishment. Then, the SMs receive the challenge (C_i) from the UC create the code/parity bit (PB_i), and send it back to the UC using the received challenge (C_i). The UC checks the PB_i. Permission is granted if it is accurate. If incorrect, the smart meter is given two more chances to respond correctly.

Gope et al. [13] have developed lightweight privacy-preserving two-factor authentication (2FA) [14] scheme based on the PUF module in IoT devices. In these schemes, PUFs are one of the factors in authenticating the devices. The BCH-based fuzzy extractor was used as an error correction solution. The fuzzy extractor removes the noise that comes during PUF's operation. The proposed scheme needs to save data on the device memory along with the public helper data, this protocol is vulnerable to physical and helper data manipulation attacks.

Alladi et al. [15] have proposed a Healthcare Authentication Protocol for achieving two-way authentication (TWA) between various entities in the Healthcare IoT environment. Healthcare authentication protocol using resource-constrained IOT devices employs challenge-response pairs generated by patient nodes and sink nodes embedded with PUF chips. After authentication between the server and the patient node, a session is established for data communication among the patient node, a sink node, and a healthcare cloud server. It takes a longer time to establish secure connections between any two entities.

Zhang et al. [16] have proposed an adaptable and successful anonymous batch authentication (ABAH) scheme to protect network security and privacy in smart vehicular networks. Based on a one-way hash chain, this scheme generates pseudo-identities for the same vehicle in vehicular networks. They offer an identity revocation scheme that allows for rapid distribution. In this scheme, the CRL size is only related to the number of revoked vehicles, regardless of the number of pseudonym certificates for revoked vehicles.

Zheng et al. [17] have presented a lightweight PUF-based two-way authentication and key exchange protocol for a decentralized model or direct IoT communication. It permits PUF-installed objects to verify one another instead of requiring internal memory of challenge–response pairs (CRPs) or any secret keys while setting up the shared key for encrypted data transmission instead of using the public-key technique. Because this protocol is developed to store information, it is insufficient for IoT devices' low-cost and resource-limited characteristics. Furthermore, stored information allows for physical attacks, proving this protocol ineffective.

Ren et al. [18] have introduced a new mutual authenticated key agreement system based on PUFs for narrow-band IoT in 5G networks. The Narrowband sources will be very reliable in 5G networks. However, much security is required, such as parallel access authentication for many devices, identity privacy protection, physical attack resistance, application traffic security, etc.

Vinoth et al. [19] have developed a new cloud-based session key agreement and data storage scheme with an enhanced authentication scheme for IoMT. These schemes obtain anonymous pre- and post-authentication. The cloud server creates a pseudo-identity in this scheme with the help of user information. The user is unaware of this false identity, which a cloud server uses to verify the identity of a user making a request. Following a successful login, the cloud server and the user share the session key, allowing data communication. This work plans to create a new model for retrieving stored documents from the cloud in the IoMT environment while minimizing searching time.

In the above literature, ROPUFs were used to generate response bits, but they produced a small number of challenge-response pairs that were predictable and consumed more power during the response bit generation process. Existing stream cipher-based encryption techniques were discussed, but these ciphers did not address data and device integrity problems, and they are also vulnerable to bit-flipping attacks. Some of the authors used a 2FA scheme between various entities in IoMT. Sometimes, the user goes through an extra step during the application login process, resulting in an extended login time.

This paper proposes the lightweight stream cipher combined with a MAC based on responses generated by an APUF to ensure data privacy protection, data integrity, and device integrity for IoMT data transmission. The scheme aims to authenticate both the sender's device and the received message using a MAC generated with the assistance of the APUF [20]. The APUF is considered a promising PUF variation for IoT devices due to its lightweight design and exponentially vast challenge-response space.

3. Preliminaries

This section discusses the descriptions of NLFSR, Hash function, hardware security, and its variants.

3.1. Nonlinear feedback shift registers (NLFSR)

The linear feedback shift register (LFSR) generated sequences have been used in many areas of a communication system, such as keystream generator in stream cipher, and pseudorandom number generator (PRNG) in cryptographic primitive methods. However, they are not cryptographically secure because the configuration of an n-bit LFSR can quickly get the result by observing 2^n contiguous bits of its sequence using the Berlekamp–Massey algorithm [21]. Because of their implicit linearity, LFSR-based stream ciphers are vulnerable to attacks, including fast linear algebraic and correlation attacks [22]. Several approaches have been proposed to overcome the linearity of the bits generated by LFSR, with one focusing on processing the output sequence of LFSRs using a nonlinear Boolean function to form a combination generator and the other on processing several bits from the LFSR state using nonlinear feedback functions; so known as NLFSR. They can be implemented in two ways: Galois and Fibonacci configurations. In Fibonacci configuration, NLFSR consists of n number of bits from 0 to $n-1$, right to left. In Fibonacci configuration, NLFSR consists of n number of bits from 0 to $n-1$, right to left. In every clock pulse, the i is moved to the bit $i-1$. The value on the 0th bit is the output of this register. The updated bit of $n-1$ is calculated by applying the feedback function of the previous state. Fig. 1 shows the Galois type of NLFSR, in which every bit of the register is updated by applying the feedback function on the previous output of the register bit. The size of the circuits implementing individual bit feedback functions is typically smaller than the depth of the circuits implementing the Fibonacci NLFSR feedback function so that propagation time can decrease. It makes Galois NLFSRs especially inspiring for cryptography applications where keystream generation speed is essential.

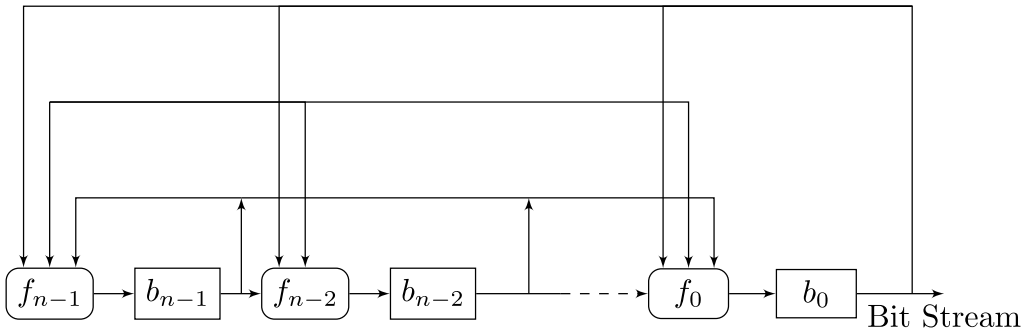


Fig. 1. Configuration of Galois NLFSR.

3.2. Neeva lightweight hash function

It is a lightweight hash function based on the construction of SPONGEN [23]. It is used in lightweight cryptography to ensure data integrity and verify authenticity. Hash functions based on the spongent construction use various parameters for different hash sizes. The parameter is N, C, R, where N is the hash size, C is the capacity, and R is the rate. The capacity of sponge-based designs decides how secure they are. Bussi et al. [24] have used Neeva lightweight hash function, where the first B bit register is taken, and $B = R + S$, where R is the rate and c is the capacity of state B. The size of state B, capacity C, and rate R is 256 bit, 224 bit, and 32 bit. The message bits are padded so that they are divided into 32 bit blocks, and this 32 bit message is XORed with the most significant 32 bits of the initial state. Initially, the register is set to all zero, that is, $S_0 = 0^r \parallel 0^c$, the first message block is XORed, and the initial state is changed. They have 16 bit words in the modified register. Then, an S-box of 4×4 on change 16-word simultaneously is applied. The first, second, and third words are then modified by XOR-ing with the fourth word while leaving the fourth word alone. To cover the bits of the full register, 64 times 4×4 S-box parallelly used. After that, an 8 bit left rotation is performed on the 256 bit register. Following that, an 8 bit left rotation is performed on the 256 bit register, followed by the addition of a round constant. After all of the register's bits have been updated, the entire process is completed in one round. Three main procedures are used to construct hash code: confusion using s-box and diffusion using XORing and rotation. Three main procedures have been used to construct hash code: confusion using s-box and diffusion using XORing and rotation. Similarly, 32 rounds are performed on whole operations.

3.3. Arbiter PUF

PUF is a hardware security primitive that uses minor variations between chips with the exact functionality to create hardware-specific identities. Such internal variances derive from unavoidable variations in the manufacturing process, and PUFs work to enhance these variations to produce responses that are unique to that hardware only. Thus, PUFs are usually represented as a challenge-response system that retrieves unique "response" data from an input "challenge". The challenge and its associated response are referred to as CRP and serve as the IC's unique identifier for that PUF only. A PUF can be categorized into two types depending on the number of CRPs it generates: weak PUFs and powerful PUFs. A weak PUF can generate a minimal number of CRPs, making it useful for creating a secret key. Due to the small number of CRPs, adversaries might be able to list every potential CRP, and CRPs must be kept secret. Strong PUF supports many CRPs, which is highly desirable for PUF-based authentication protocols. Arbiter PUF [25,26] is a type of strong PUF extensively utilized in IoT devices.

4. Proposed scheme

In this section, the detail of the proposed scheme is discussed in detail. The proposed scheme consists of three modules; (1) Configuration of a new NLFSR-based keystream generation algorithm (2) the design of PUF, and (3) the Generation of MAC.

The detailed configuration of the NLFSR is discussed in Section 4.1. The NLSFR is used for two purposes (i) Generation of challenge x_i for the APUF and (ii) generation of keystream k_i for the stream cipher. As this is a symmetric cryptosystem both the received and sender share the same secret key K which is the seed for NLSFR. The NLSFR generates the input challenge x_i, which is given as input to the APUF, and the APUF generates the corresponding response bits r_i as shown in 2. The detailed configuration of the PUF is discussed in Section 4.2.

A Neeva lightweight hash function-based MAC generator is used which is discussed in Section 4.4. The message m_i along with the PUF response r_i are used to produce the MAC value mac_i.

The sender uses the NLSFR to generate the challenge x_i for the APUF and (ii) keystream k_i for the stream cipher. The challenge x_i is passed to the PUF to generate response r_i which is concatenated with the message m_i and provided as input to the MAC generator. The MAC generator produces the corresponding MAC value mac_i. On the other hand, the sender encrypts the message m_i using a

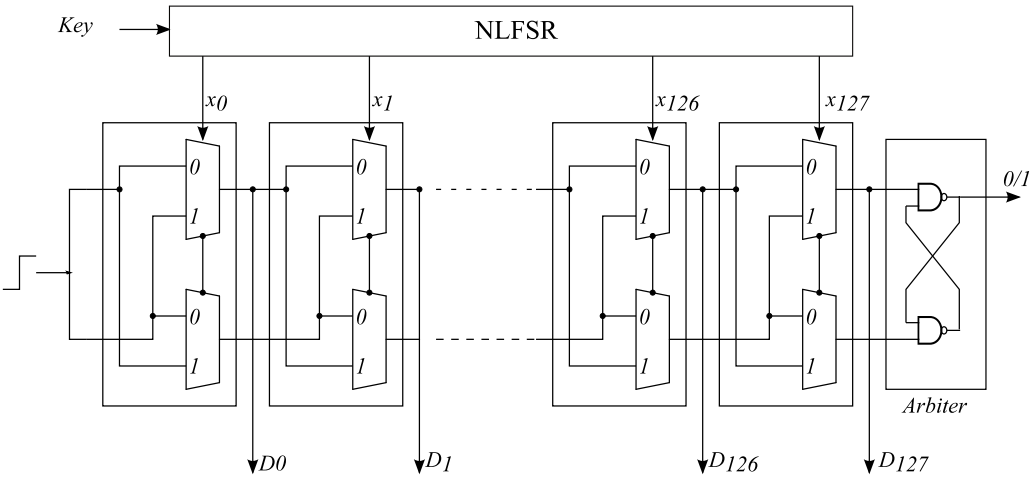


Fig. 2. Design of the physically unclonable function (PUF).

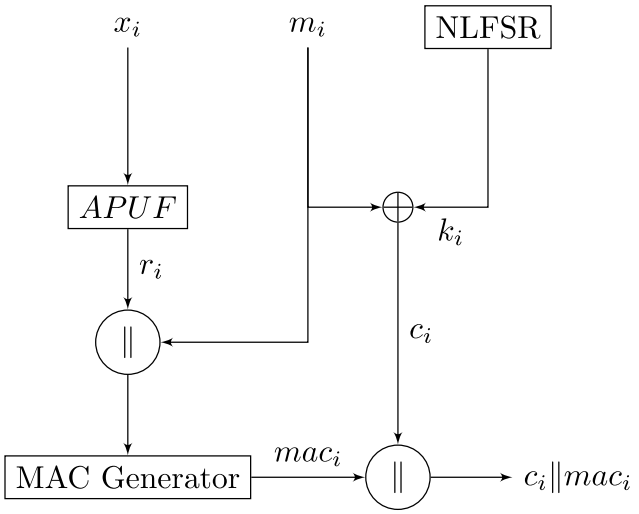


Fig. 3. The overall process at the sender side.

key stream k_i generated by the NLSFR and generates the corresponding ciphertext c_i. Finally, the MAC value mac_i is appended with the ciphertext c_i, and the value $c_i \parallel mac_i$ is transmitted to the receiver. The overall process at the sender side is presented in Fig. 3.

The receiver receives the value $c_i \parallel mac_i$ and extracts the MAC value mac_i and the ciphertext c_i. Then it decrypts the c_i first using a key stream k_i generated by the NLSFR and finds the message m_i. After that, the receiver retrieves the APUF response r_i' from the challenge-response table for the corresponding challenge x_i. The response r_i' is concatenated with the message m_i and provided as input to the MAC generator. The MAC generator produces the corresponding MAC value mac_i'. Finally, the received compares the received MAC value mac_i with the generated MAC value mac_i'; if they match then the receiver will confirm that the sender device is authenticated and accepts the message. The overall process at the receiver side is presented in Fig. 4.

4.1. Design of NLFSR-based lightweight encryption

This section discusses the detailed design of an NLFSR-based cipher. This cipher consists of three different blocks named $NLFSR_1$, $NLFSR_2$, and the output generation function $Z(x)$, and each NLFSR has an equal size of 128 bits. The initial value of $NLFSR_1$ is denoted as $s_i, \ldots, s_{i+127}$ and the initial value of $NLFSR_2$ is denoted as $s_{i+127}, \ldots, s_{i+255}$. The design of this cipher is shown in Fig. 5. In Fig. 5, $f_L(x)$ represents the linear function and is calculated by Eq. (1), while $f_N(x)$ is the nonlinear function calculated by Eq. (2).

$$f_L(x) = 1 + x^{14} + x^{48} + x^{112} + x^{120} + x^{124} \tag{1}$$

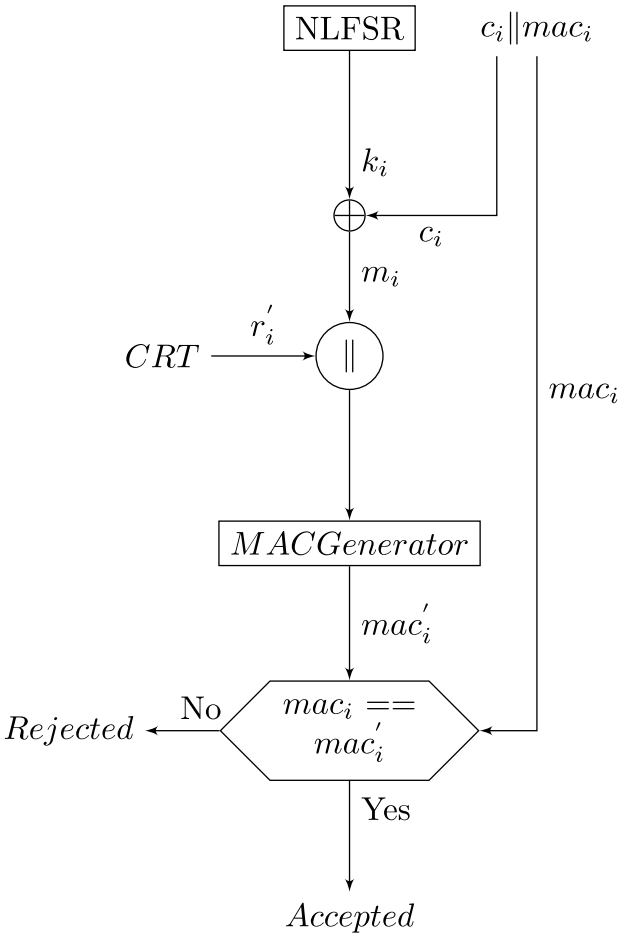


Fig. 4. The overall process at the receiver side.

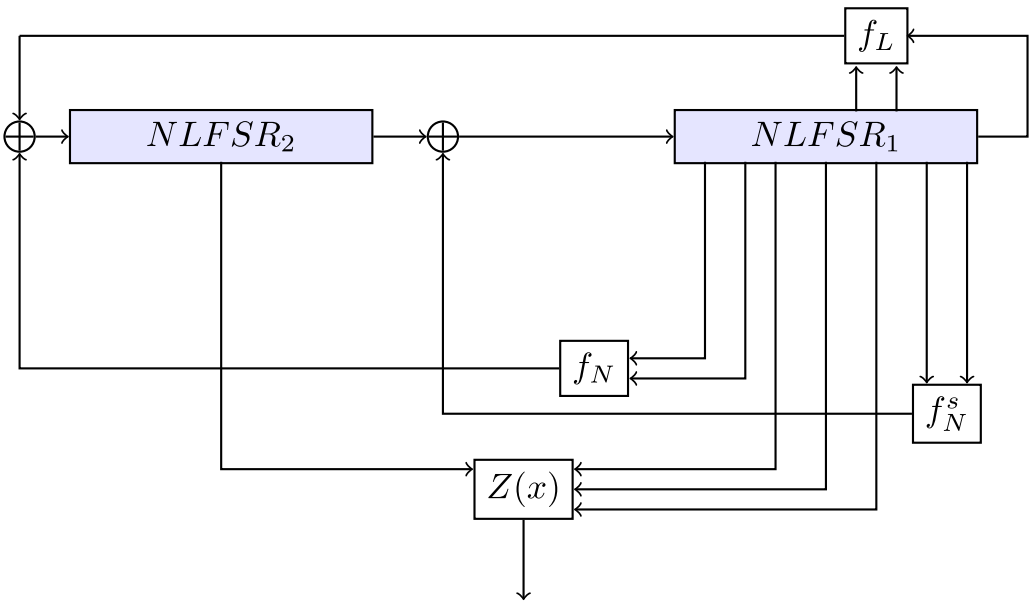


Fig. 5. The structure of NLFSR-based key stream generator.

Algorithm 1 KSGA

procedure KSGA($k1, k2$)
  $x_{36}, x_{45}, x_{63}, x_{64}, x_{70} \leftarrow NLFSR_1(k1)$
  $x_{192} \leftarrow NLFSR_2(k2)$
  $f^1 \leftarrow f_L(x_0, x_{12}, x_{48}, x_{115}, x_{117}, x_{127})$
  $f^2 \leftarrow f_N(x_{41}, x_{70}, x_{46}, x_{87}, x_{52}, x_{110}, x_{55}, x_{113}, x_{117}, x_{110}, x_{120}, x_{127})$
  $IP_{fb1} \leftarrow f^1 \bigoplus f^2$
  $I_{fb2} \leftarrow x_{129} \bigoplus f_N^s$
  $KS \leftarrow XOR(x_{36}, x_{45}, x_{63}, x_{64}, x_{70}x_{192}, x_{129})$
end procedure

Algorithm 2 Encryption Process

procedure ENCRYPTION($k1, k2, m$)
  k ←KSGA(k1,k2)
  for $i = 1$ to N **do**
    $C_i \leftarrow m_i \bigoplus k_i$
  end for
  return C
end procedure

$$f_N(x) = 1 + x^{31}x^{68} + x^{39}x^{84} + x^{62}x^{111} + x^{65}x^{114} + x^{67}x^{116} + x^{76}x^{123} + x^{91}x^{124}x^{135}x^{138} \tag{2}$$

The feedback polynomial of $NLFSR_2$ is denoted by

$$f_{255}(x) = 1 + f_L(x) + f_N(x) \tag{3}$$

The feedback polynomial of $NFSR_1$ is defined by,

$$f_{128}(x) = x^{129} \oplus f_N^s(x) \tag{4}$$

$$f_N^s(x) = 1 + x^4x^{41} + x^{12}x^{57} + x^{35}x^{84} + x^{38}x^{87} + x^{40}x^{89} + x^{49}x^{96} + x^{64}x^{97}x^{108}x \tag{5}$$

where f_N^s represents the nonlinear terms of the feedback polynomial of $NLFSR_1$, which are obtained from the shifted positions of the nonlinear terms' variable in the feedback polynomial of $NLFSR_2$.

$$f_{255}(x) = f_L(x) \oplus f_N(x) \tag{6}$$

where $f_L(x)$ is a linear polynomial function of six variables from $NLFSR_1$ and $f_{255}(x)$ of F is of the type is a linear in which $f_L(x)$ function of six variables and $f_N(x)$ is a nonlinear function of twelve variables. The contents of these two shift registers represent the cipher's state. Five independent variables are fed into the Boolean function $Z(x)$. Four variables are taken from $NLFR1$, while one variable is taken from $NLFSR2$. This function is represented as

$$Z(x) = x_{70} + x_{64} + x_{63}x_{45} + x_{63}x_{64} + x_{45}x_{70} + x_{70}x_{63}x_{64} + x_{36}x_{45}x_{64} + x_{70}x_{45}x_{64} + x_{70}x_{63}x_{64}x_{192} \tag{7}$$

where the variables $x_{36}, x_{45}, x_{63}, x_{64}, x_{70}$ corresponds to the tap positions $s_{i+35}, s_{i+44}, s_{i+62}, s_{i+63}, s_{i+69}$ and s_{i+191} respectively. The output of the filter function is keystream (k_i), which has been used in Fig. 3 to encrypt the medical dataset.

4.2. Configuration of APUF

Arbiter PUF is based on the switch, 2-multiplexer used in a switch and gives the same input to both. This multiplexer's output is determined by the selection line, and the selection line is considered an input challenge of this multiplexer. Two delay lines are built into it as a series of switching blocks, each having an arbiter block at the end. Every switch block consists of three input lines and two output lines. The first input of each switch is connected to enable signals, and each block's output is connected to the next switch block serially. The outputs of the final switching blocks go to the arbiter block and select which signals arrive first. Based on this outcome, the arbiter creates a single bit known as the response bit. A 128 bit Arbiter PUF is shown in Fig. 2, and it contains two parallel 128 multiplexer switches connected serially. The first input that connects to the enable signal is given as the first input

to both multiplexers. The D-flip flop generates a 128 bits input challenge $x_0, x_1, \ldots, x_{127}$. The select line of the multiplexer is used as an input challenge, which ultimately selects a double multiplexer line whose propagation of the input signals is at the two input terminals.

The arbiter at the end of two paths determines which path is faster. The first multiplexer's two input terminals are coupled to a common end, and the input pulse to the entire PUF is applied to this common terminal. The arbiter announces the winning route at the finish line and encodes the result as a single bit of output response (logic-0 or logic-1). A total of 2^{128} possible pairs can be chosen for the 128 challenge bits $x_0, x_1, \ldots, x_{127}$, with each bit controlling two multiplexer-switching components. There is one specific path pair for every applied challenge. Let t_1 and t_2 be the propagation delays of the trigger pulse across the two distinct channels determined by the challenged NAND input and clock of the arbiter, respectively. Let the two paths be determined by applying the 128 bit challenge $x_0, \ldots, x_{127}$ to the APUF. The response produced by the APUF circuit is then delivered by:

$$r = \begin{cases} 1, & \text{if } t_1 < t_2 \\ 0, & Otherwise \end{cases}$$

4.3. Fuzzy extractor for PUF response

Fuzzy Extractor (FE) is a popular coding technique for removing response noise and correcting errors. It is used to extract random bits from noisy and unevenly distributed data. FE is made up of two methods: Generation (GEN) and Reproduction (REP). The GEN algorithm takes the original response (r) as input and generates uniformly random bits (Rb) that can be used as secret keys or non-secret strings. The REP algorithm takes two inputs, helper data and w', and reproduces r' from the noisy response (w'). The reproduction process is only successful if r and r' are close enough.

4.4. MAC generator

The Neeva hash function is used to generate the MAC values. Here, r_i is the response bit generated from PUF, and the total number of responses is 128. M is the single iteration input message which has a size 192 bit. For the sponge-based hash function, The Neeva hash function requires the input message block size to be 32 bits. Divide the 192 bit message into six blocks following the spanned-based hash function, with each message block 32 bits.

$$M = r_i \parallel m_i$$

$$M = M_1 \parallel M_2 \parallel M_3 \parallel M_4 \parallel M_5 \parallel M_6$$

Each M_i is a 32 bit block required for the hash function.

$$mac_i = H_{Neeva}(M)$$

For authentication of the sender device and the data, a PUF-based MAC is used. The designed PUF generated responses r_i against corresponding challenges x_i. To generate the MAC mac_i, the data m_i and the response r_i are concatenated, and the hash H is generated, which is treated as mac_i. The sender device will generate keystream k_i using the designed NLFSR. The data m_i will be encrypted first, producing the cipher stream $c_i = m_i \oplus k_i$.

5. Experiment result and security analysis

In this section, evaluate the randomness properties of the ciphertext bits and ensure the security of the proposed algorithm. Security analysis primarily focuses on identifying and mitigating security risks, ensuring data confidentiality, integrity, and availability, and assessing compliance with security standards.

5.1. Experiment layout

Dataset: The experiment is performed on the medical dataset named "Heart Disease Dataset" taken from Kaggle. The dataset contains more than 10 thousand records and 14 attributes. The size of each record contains approximately 112 bits. Each record in the dataset has an approximate size of 112 bits.

Baseline approaches : Four baselines are used to compare with the proposed algorithm,

In RC4-1 [27], the basic RC4 algorithm is modified, and the KSA+ algorithm improves security without increasing encryption time. The KSA+ algorithm modifies the mathematical computations and adds more randomness.

In A5/1+ [28], the algorithm uses an NLFSR to generate the keystream. Modifications have been made to enhance the randomness properties of the basic A5/1 algorithm, making it more robust against attacks.

In RCCM [7], the chaotic map is used in the Key Scheduling Algorithm (KSA) and the Pseudo-Random Number Generator (PRNG) algorithm of RC4 to generate the keystream. This modification in RC4 has improved the randomness properties of the keystream.

In PARCCM [7], product algebra is used to improve the RCCM and it could also improve the randomness and periodicity of the keystream.

Experimental Tools All experimental data are implemented in Python and the MATLAB R2020b programming language. The most popular real dataset for testing is selected.

Table 1
NIST STS tests performed on generated ciphertext bits when ciphertext bit length is 10^6 bits.

	RC-4	A5/1+	RCCM [7]	PARCCM [7]	Proposed Scheme
Test name	P-value	P-value	P-value	P-value	P-value
Monobit test	0.3565	0.4099	0.7233	0.9700	0.6467
Frequency test	0.4360	0.4750	0.5057	0.9800	0.7178
Runs test	0.4347	0.5099	0.5230	1.0000	0.5518
Test for the longest-run-of-ones in a block	0.4906	0.5072	0.4919	1.0000	0.9205
Binary matrix rank test	0.5356	0.5069	0.5167	0.9800	0.3879
Discrete fourier transform test	0.4575	0.5158	0.5268	0.9910	1.0000
Non-overlapping template matching test	1.0000	0.9999	1.0000	1.0000	0.6283
Overlapping template matching test	0.5343	0.5078	0.5382	0.9801	0.8330
Maurer's 'Universal Statistical' test	0.9987	0.9989	0.9999	1.0000	0.0856
Linear complexity test	0.5510	0.5643	0.5215	1.0000	0.4768
Serial test	0.1859	0.4349	0.4334	0.9911	0.2712
Approximate entropy test	0.0000	0.0000	0.4864	0.9710	0.2269
Cumulative sums test	0.3121	0.4456	0.4518	0.9711	1.0000
Random excursions test	0.1640	0.1311	0.1596	0.9320	0.4283
Random excursions variant test	0.0741	0.0819	0.0712	0.8135	0.5589

5.2. NIST STS

The NIST STS [29] is used in the security community to test various randomness properties of PRNGs. The NIST STS contains 15 tests, the P-value and Proportion are two indicators for each test item's outcomes. The sequence is random if the test sequence value is greater than the significance level β, and $\beta \in [0.001, 0.01]$. The larger the P-value, the more unpredictable the test sequence. The significance level used by NIST STS is directly related to the percentage of sequences that pass a test. If the test sequence's outcome falls inside the confidence interval value, the test. More crucially, the Monobit Test is the most fundamental approach out of the 15 test, and all subsequent tests are based on it. The sequence being tested is random if the Monobit test succeeds. If the Monobit test fails, the next tests are pointless. The test results are compared and presented in Table 1. Bassham et al. [29] provide a more detailed explanation of the 15 tests. The algorithm generates encrypted medical dataset bits with a length of 10^6. However, the tests have a small impact on the randomness of the methods. The A5/1+ and RC4-1 algorithms in Table 1 failed in Approximate Entropy Testing. This test aims to compare the frequency of overlap of the image data between neighboring blocks to the predicted outcome of a random sequence.

The following subsection provides a security evaluation of each presented attack scenario.

5.3. Security analysis

This section presents the security analysis of the proposed encryption method along with the data and device integrity techniques.

5.3.1. Physical attack

Attackers try to access the device and retrieve the private data stored in local memory. A PUF-based MAC is used in which challenges are given to the PUF and generate response keys as an output of PUF; therefore, keeping the secret key in local memory is unnecessary. Because no data is held there, attackers cannot steal any data from memory, trying to make the proposed protocol resistant to physical attacks.

5.3.2. Modeling attack

An attacker attempts to obtain CRPs of the deployed PUF from the server and IoMT devices. The machine learning techniques are then applied to the CRP dataset, and the attacker can guess the acceptable response to a particular challenge. The proposed system uses PUF-based MAC authentication, and the direct responses are not exchanged across an insecure channel. Thus, the attacker cannot intercept the messages transferred between the IoMT object and the server. So, a modeling attack is not possible on the proposed authentication protocol.

5.3.3. Replay attack

The attackers in this attack kept the communicated information from the trusted and wide procedure to use it in later authentication. To prevent this attack, an encrypted message with a MAC is sent, which prohibits the attackers from stealing original information from the communication channel for future communication. The proposed method also appends factors, such as time to transfer messages using both IoMT devices and the authorized server, that help protect against this assault. During each authentication phase, the user changes the challenge, response, private key, and session key. Therefore, the attack will be impossible.

5.3.4. Cloning attack

This attack is possible on PUF based on memory. It is possible by tinkering on the back side of the circuit, calling its unclonability into doubt and exposing the door to imitative in an untrustworthy supply chain. An attacker [30] has used PES and FIB techniques to reveal this attack on SRAM PUF. These techniques are used in semiconductor devices, materials research, and increasingly in the biological area. The PES determines the PUF value based on information provided by photons emitted from the reverse side. In this paper, An Arbiter PUF is used; each value from a PUF output is distinct, and cloning a PUF is challenging since each variation to the physical features of the PUF circuits creates a different version of the PUF, resulting in answers that differ from the initial PUF circuit.

5.3.5. Helper data manipulation

By changing the auxiliary data transmitted or stored on the device memory, this threat attempts to render fuzzy extractor operations infeasible. This technique [31] is safely generated by the server and sent to the IoMT devices. A MAC value and other metadata are sent along with each transmission of helper data to allow the IoT side to verify the integrity of the helper data. Also, the modification must be performed when the information is sent to the IoMT device. This attack is impossible because messages use MAC based on PUF. By changing the auxiliary data transmitted or stored on the device memory, this attack attempts to render fuzzy extractor operations infeasible.

5.3.6. Message authentication attack

Pseudo-Collision Resistance: In this attack, the attacker received MAC from MAC associated with data, and then they tried to alter the MAC value using a compression function. The Neeva hash function is used to generate the MAC value, and this hash function resists such attacks because it is a challenging authentication scheme in cryptography. The input passed to the Neeva hash function is unpredictable or random enough that the attacker cannot break the MAC value.

Resistance to Birthday Attacks: This hash function tries to break the hash algorithm by taking two first identical messages with MAC value in less than $2^{N/2}$ trials (where the N is the size of the message and also equal to the size of the MAC value). The MAC size in the developed scheme is 128, which needs $2^{N/2}$ trails for brute force attack.

5.3.7. Key space analysis

The set of all possible keys generated by the designed LFSR and MAC secret key is reasonably large. It is equal to 2^r and 2^k, where $r = 128$ and $k = 256$. According to the keyspace analysis study [32], keyspace should be greater or equal to 2^{128} to prevent a brute force attack. The size of the key space is 2^{384} bits, these numbers are large enough to resist the brute force attack.

5.4. Side-channel attacks

Side-channel attacks are a significant security concern for IoMT devices, as they can exploit unintentional information leakage through various channels like power consumption, electromagnetic radiation, and timing. To mitigate side-channel attacks on IoMT devices, a highly secure NLFSR-based stream cipher encryption algorithm is proposed. This cipher protects the encryption key from side-channel attacks. PUF-based hardware security protects IoMT devices from side-channel attacks by generating unique cryptographic keys based on physical variations, making it extremely difficult for attackers to extract keys through side-channel analysis. This dynamic key generation enhances device security and mitigates the risk of key leakage.

6. Conclusion

The encryption is done by NLFSR-based stream cipher to ensure the privacy preservation of the IoMT data. An Arbiter PUF is used to generate a large number of CRPs, as shown in Fig. 2. Then, the Neeva hash function is applied to generate the MAC among generated response bits and message bits. The MAC ensures data and device integrity. To verify the randomness properties of ciphertext bits, apply the NIST STS on the ciphertext bits. The presented PUF-based MAC and NLFSR-based encryption techniques are used for encrypting the health monitoring dataset (referred to as the dataset) with IoMT devices. This proposed scheme does not require any local memory device to hold the information, preventing various attacks, particularly physical ones. The security analyses are done to verify that our designed method is resistant to most existing tactics, notably physical attacks. In the future, the proposed scheme may be tested on a large IoMT network to realize its real-time performance.

CRediT authorship contribution statement

Punam Kumari: Data curation, Formal analysis, Investigation, Methodology, Visualization, Writing – original draft. **Bhaskar Mondal:** Investigation, Methodology, Project administration, Resources, Supervision, Validation, Writing – review & editing.

Declaration of competing interest

The authors certify that they have NO affiliations with or involvement in any organization or entity with any financial interest (such as honoraria; educational grants; participation in speakers' bureaus; membership, employment, consultancies, stock ownership, or other equity interest; and expert testimony or patent-licensing arrangements), or non-financial interest (such as personal or professional relationships, affiliations, knowledge or beliefs) in the subject matter or materials discussed in this manuscript.

Data availability

No data was used for the research described in the article.

References

[1] H. Zhang, J. Chen, L.Y. Zhang, C. Fu, R. Gravina, G. Fortino, Z. Lv, Low-cost and confidential ecg acquisition framework using compressed sensing and chaotic systems for wireless body area network, IEEE J. Biomed. Health Inform. 26 (12) (2022) 5783–5792.

[2] G. Wu, S. Wang, Z. Ning, B. Zhu, Privacy-preserved electronic medical record exchanging and sharing: A blockchain-based smart healthcare system, IEEE J. Biomed. Health Inf. 26 (5) (2021) 1917–1927.

[3] M.A. Qureshi, A. Munir, Puf-rake: A puf-based robust and lightweight authentication and key establishment protocol, IEEE Trans. Dependable Secure Comput. 19 (4) (2021) 2457–2475.

[4] H. Aziza, J. Postel-Pellerin, H. Bazzi, P. Canet, M. Moreau, V. Della Marca, A. Harb, True random number generator integration in a resistive ram memory array using input current limitation, IEEE Trans. Nanotechnol. 19 (2020) 214–222.

[5] H. Hu, Q. Chen, J. Xu, B. Choi, Assuring spatio-temporal integrity on mobile devices with minimum location disclosure, IEEE Trans. Mob. Comput. 16 (11) (2017) 3000–3013.

[6] H. Li, V. Kumar, J.-M. Park, Y. Yang, Cumulative message authentication codes for resource-constrained IoT networks, IEEE Internet Things J. 8 (15) (2021) 11847–11859.

[7] T. Liu, Y. Wang, Y. Li, X. Tong, L. Qi, N. Jiang, Privacy protection based on stream cipher for spatiotemporal data in IoT, IEEE Internet Things J. 7 (9) (2020) 7928–7940.

[8] G. Zeng, X. Dong, J. Bornemann, Reconfigurable feedback shift register based stream cipher for wireless sensor networks, IEEE Wirel. Commun. Lett. 2 (5) (2013) 559–562.

[9] S.A. Parah, J.A. Kaw, P. Bellavista, N.A. Loan, G.M. Bhat, K. Muhammad, V.H.C. de Albuquerque, Efficient security and authentication for edge-based Internet of Medical Things, IEEE Internet Things J. 8 (21) (2020) 15652–15662.

[10] V.P. Yanambaka, S.P. Mohanty, E. Kougianos, D. Puthal, Pmsec: Physical unclonable function-based robust and lightweight authentication in the internet of medical things, IEEE Trans. Consum. Electron. 65 (3) (2019) 388–397.

[11] Z. Guan, H. Liu, Y. Qin, Physical unclonable functions for IoT device authentication, J. Commun. Inf. Netw. 4 (4) (2019) 44–54.

[12] M. Mustapa, M.Y. Niamat, A.P.D. Nath, M. Alam, Hardware-oriented authentication for advanced metering infrastructure, IEEE Trans. Smart Grid 9 (2) (2016) 1261–1270.

[13] P. Gope, B. Sikdar, Lightweight and privacy-preserving two-factor authentication scheme for IoT devices, IEEE Internet Things J. 6 (1) (2018) 580–589.

[14] M. Kaveh, S. Aghapour, D. Martin, M.R. Mosavi, A secure lightweight signcryption scheme for smart grid communications using reliable physically unclonable function, in: 2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe, EEEIC/I & CPS Europe, IEEE, 2020, pp. 1–6.

[15] T. Alladi, V. Chamola, et al., Harci: A two-way authentication protocol for three entity healthcare IoT networks, IEEE J. Sel. Areas Commun. 39 (2) (2020) 361–369.

[16] J. Zhang, H. Zhong, J. Cui, Y. Xu, L. Liu, An extensible and effective anonymous batch authentication scheme for smart vehicular networks, IEEE Internet Things J. 7 (4) (2020) 3462–3473.

[17] Y. Zheng, W. Liu, C. Gu, C.-H. Chang, Puf-based mutual authentication and key exchange protocol for peer-to-peer IoT applications, IEEE Trans. Dependable Secure Comput. 20 (4) (2023) 3299–3316, http://dx.doi.org/10.1109/TDSC.2022.3193570.

[18] X. Ren, J. Cao, M. Ma, H. Li, Y. Zhang, A novel puf-based group authentication and data transmission scheme for nb-IoT in 3gpp 5 g networks, IEEE Internet Things J. 9 (5) (2021) 3642–3656.

[19] R. Vinoth, L.J. Deborah, P. Vijayakumar, B.B. Gupta, An anonymous pre-authentication and post-authentication scheme assisted by cloud for medical IoT environments, IEEE Trans. Netw. Sci. Eng. 9 (5) (2022) 3633–3642.

[20] M.H. Mahalat, S. Mandal, A. Mondal, B. Sen, An efficient implementation of arbiter puf on fpga for IoT application, in: 2019 32nd IEEE International System-on-Chip Conference, SOCC, IEEE, 2019, pp. 324–329.

[21] H. Ali, G.M. Fathy, Z. Fayez, W. Sheta, Exploring the parallel capabilities of gpu: Berlekamp-massey algorithm case study, Cluster Comput. 23 (2) (2020) 1007–1024.

[22] Y. Wei, E. Pasalic, Y. Hu, A new correlation attack on nonlinear combining generators, IEEE Trans. Inform. Theory 57 (9) (2011) 6321–6331.

[23] A. Bogdanov, M. Knezevic, G. Leander, D. Toz, K. Varici, I. Verbauwhede, Spongent: The design space of lightweight cryptographic hashing, IEEE Trans. Comput. 62 (10) (2012) 2041–2053.

[24] K. Bussi, D. Dey, M. Kumar, B. Dass, Neeva: A lightweight hash function, Cryptology ePrint Archive, 2016.

[25] D.P. Sahoo, D. Mukhopadhyay, R.S. Chakraborty, P.H. Nguyen, A multiplexer-based arbiter puf composition with enhanced reliability and security, IEEE Trans. Comput. 67 (3) (2017) 403–417.

[26] C. Herder, M.-D. Yu, F. Koushanfar, S. Devadas, Physical unclonable functions and applications: A tutorial, Proc. IEEE 102 (8) (2014) 1126–1141.

[27] P. Jindal, S. Makkar, Modified rc4 variants and their performance analysis, in: G. Panda, S.C. Satapathy, B. Biswal, R. Bansal (Eds.), Microelectronics, Electromagnetics and Telecommunications, Springer Singapore, Singapore, 2019, pp. 367–374.

[28] D. Upadhyay, P. Sharma, S. Valiveti, Randomness analysis of a5/1 stream cipher for secure mobile communication, Int. J. Comput. Sci. Commun. 3 (2014) 95–100.

[29] L.E. Bassham, A.L. Rukhin, J. Soto, J.R. Nechvatal, M.E. Smid, E.B. Barker, S.D. Leigh, M. Levenson, M. Vangel, D.L. Banks, et al., Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications, 2010.

[30] S.B. Dodo, R. Bishnoi, S.M. Nair, M.B. Tahoori, A spintronics memory puf for resilience against cloning counterfeit, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 27 (11) (2019) 2511–2522.

[31] J. Delvaux, D. Gu, D. Schellekens, I. Verbauwhede, Helper data algorithms for puf-based key generation: Overview and analysis, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 34 (6) (2014) 889–902.

[32] C. Tezcan, Key lengths revisited: Gpu-based brute force cryptanalysis of des, 3des, and present, J. Syst. Archit. 124 (2022) 102402.