2021 IEEE International Conference on Omni-Layer Intelligent Systems (COINS) | 978-1-6654-3156-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/COINS51742.2021.9524282

An Error Correction Approach to Memristors PUF-based Key Encapsulation

Ashwija Reddy Korenda, Sareh Assiri, Fatemeh Afghah, and Bertrand Cambou
School of Informatics, Computing and Cyber Systems, Northern Arizona University Flagstaff, AZ. 86011
{ashwijakorenda, sa2363, fatemeh.afghah, bertrand.cambou}@nau.edu

***Abstract*—A novel Physically Unclonable Functions (PUF)-based keyless encapsulation protocol was recently proposed by our team that does not rely on cryptographic keys and directly encrypts the messages using the unique responses extracted from embedded ReRAM PUFs in the devices. Therefore, this method avoids challenges and security threats due to key storage and distribution. Since this protocol uses analog resistance values from ReRAM PUFs for message encryption, the protocol is sensitive to variations in PUF responses due to environmental conditions. Therefore, it requires schemes to minimize the impact of such variations to guarantee the integrity of the decryption of the message. In this paper, we develop protocols to utilize Error Correction Coding (ECC) mechanisms to stabilize the cipher text encrypted with this protocol. The ECC-based key encapsulation protocol was evaluated using Reed-Solomon and BCH codes. The results show the effectiveness of the proposed protocol in offering noise-free messages under common ranges of environmental variations in ReRAM PUF responses.**

***Keywords*— Key encapsulation, Keyless Encryption, Physically Unclonable Functions, memristor PUFs, error correction.**

I. Introduction

Most cryptography schemes use cryptographic keys to encrypt the messages. These schemes rely on reliable mechanisms for key generation, key distribution, and storage of these keys, which lead to increased complexity in power-constrained networks. These requirements would be particularly challenging for IoT networks which consists of billions of nodes, some with very low power. Keyless Encryption is an attractive choice for low-power IoT devices with limited memory, as it eliminated the need to store the keys, as well as the need for the generation and distribution of billions of secret keys among these devices. Moreover, the proposed keyless encryption is not vulnerable against physical attacks attempting to extract the secret keys stored in non-volatile memory of IoT devices. As we eliminate the need for keys to encrypt messages, the problems related to scalability of keys in key-based methods are automatically mitigated.

Recently, a keyless encryption protocol using memristor as Physically Unclonable Functions (PUF) was developed based on the idea that the injection of low currents in cells of memristor arrays can result in ephemeral conductive paths, and stochastic resistances [1]. In this protocol, the plain-text (PT) is integrated as a part of resistance values obtained from the memristor PUF at a particular cell address, where a certain current is injected to create a cipher text. However, the memristor PUFs are subject to variations in resistance measurements due to the natural drifts of physical parameters with time and environmental conditions (e.g. temperature, humidity, aging, background noise) which make the keyless encryption protocol challenging.

In this paper, we propose utilization of Error Correction Codes (ECC) to facilitate the operation of such keyless encryption in practical settings by correcting a portion of errors caused by the noise in PUF responses. There has been a rise in attacks by analyzing the differential power dissipated in cryptography to extract keys. This type of attack is very practical and non-invasive, where the hackers are able to extract the keys by analyzing the differential power [2], [3]. As the IoT devices have a limitation on power and memory for most devices [4]–[6], it is harder to implement complex schemes using long-secret keys and strong cryptographic schemes. Therefore, the proposed keyless encryption has the potential to be a safer encryption method for IoTs.

The rest of this paper is organized as follows: In Section II, we present background information about the PUFs, resistive RAMs, ECC and the recently developed keyless protocol. In Section III, we introduce the ECC-based keyless encryption method. In Section IV, we present the experimental results followed by the conclusions.

II. BACKGROUND

Recently several research works focused on the use of error correction code with PUFs [7]–[11]. The authors in [8], proposed a new index based syndrome coding scheme to ensure the stability of PUF keys while limiting the information leakage of helper data. Several Fuzzy Extractor (FE) based PUF-based key generation mechanisms where helper data string consisting of code-offset and syndrome based FE were discussed and compared them to existing methods [12], [13]. The generated helper data is later used for reliable reconstruction of the initial key. [12] discusses the current FE schemes and compares their performance on Xilinx FPGAs. [13] proposes masking of linear ECCs while ensuring their correction capability is not compromised when utilized in PUFs. To the best of our knowledge, all these works focused on encryption of data using cryptographic keys generated from PUFs. We concentrate on

This material is based upon the work supported by the National Science Foundation under Grant No. 1827753.

how to utilize ECCs for keyless encryption of messages using a PUF. In this section we provide a brief description of memory-based PUF technologies followed by an introduction to the keyless encryption protocol and some common ECC schemes.

A. Physically Unclonable Functions (PUFs)

Different PUF technologies have been utilized to generate several cryptographic primitives including cryptographic keys and random numbers in various security applications such as identification, authentication, generation of cryptographic keys and generation of random numbers [14]. The PUFs need to be unclonable with high levels of entropy making them statistically unique. Various types of PUF have been recently utilized in security applications including ring oscillators, and memory structures such as SRAM, DRAM, Flash, ReRAM, and MRAM [15]. The output of PUF in response to a specific input-*challenge*, is called as the *response*. The response generated upfront from the PUF during the enrolment is called the *'original response'*, whereas the responses generated during the access control rounds or authentication rounds are referred to as *'regenerated or noisy responses'* or simply called as responses. As a PUF is based on physical characteristics of a device, the responses can be sensitive to aging, and environmental conditions such as temperature, voltage drifts or electrostatic interference. These variations hamper the performance of the PUF-based key generation and keyless encryption methods.

1) Memristor PUF: The concept of the memristor was introduced by Chua [16] in 1971, a missing circuit element which operates as both a memory and a transistor. It was then proposed that the flux of charge injected through the device during a fixed period of time can minimize the value of the resistance. In 2017 according to [17], it was observed that the injection of low currents in cells of memristor arrays can result in dissolvable conductive paths of variable resistances, and can be exploited to design PUFs. An example of the memristor is the Resistive Random-Access Memory (ReRAM). The ReRAM has shown a great potential as one of the most efficient memory technologies, with its unique features such as high density, low-power, and non-volatility. In [18], the randomness and stability of the ReRAM PUF have been measured, where the results show that the robustness of the ReRAM PUF against spatio-temporal variations was significant (close to 50%). Hence, in this study, the ReRAM has been selected as the PUF, where the challenge is implemented by injecting different low current values into different ReRAM cells and the resistance of these cells is considered as the response.

2) Addressable PUF generators (APG) : We utilize the concept of Addressable PUF generator (APG) proposed in [19] to extract a response from memory-based PUFs based on an address. The original responses are stored in the server as a reference. This mechanism allows us to arrive at a particular cell address by using a password and a random number where the random number is shared between the server and client using handshake mechanisms and the password is already known to both the client and server. Therefore, allowing both parties to arrive at the same PUF address.

B. Memristors PUF-based Keyless Encryption

The keyless protocol is based on encrypting messages using memristor arrays of cells; where a set of cells located at different addresses are driven at various levels of currents. This protocol uses multi-factor authentication, and approaches like APGs to extract the PUF response which is integrated with the plain text to generate a cipher text.

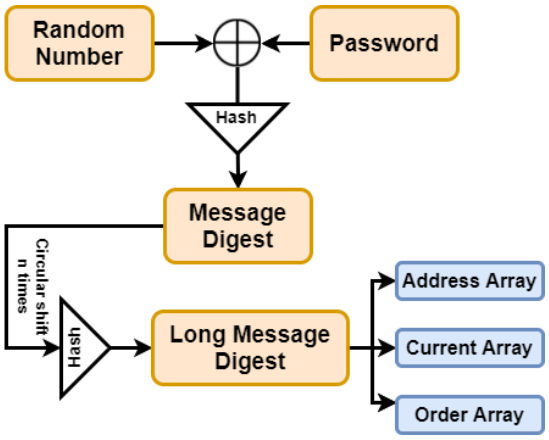


Fig. 1: Extraction of Address, Current and Order Array from LMD using APG Protocol

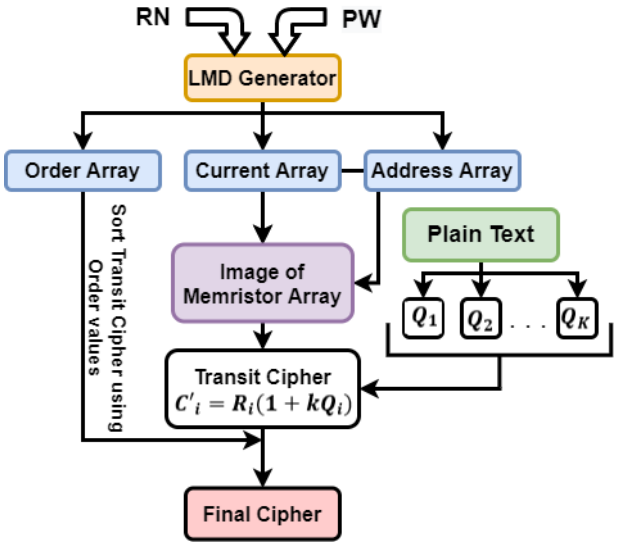


Fig. 2: Flowchart detailing the encryption process using the memristor PUF [20].

Figure 2 details all the protocol steps in memristor PUF-based keyless encryption. In this protocol, a password (PW) and a random number (RN) are XORed and then hashed to get a message digest (MD). The MD is then sent to an eXtended Output Function (XOF) which allows us to extend our MD to a desired length in order to extract a Long Message Digest (LMD). This LMD is divided into the address, current and ordered arrays as shown in Figure 1. During the encryption, the plain text in ASCII format is divided into 4-bit symbols, whose decimal equivalent ranges from 0 to 15. Each symbol is

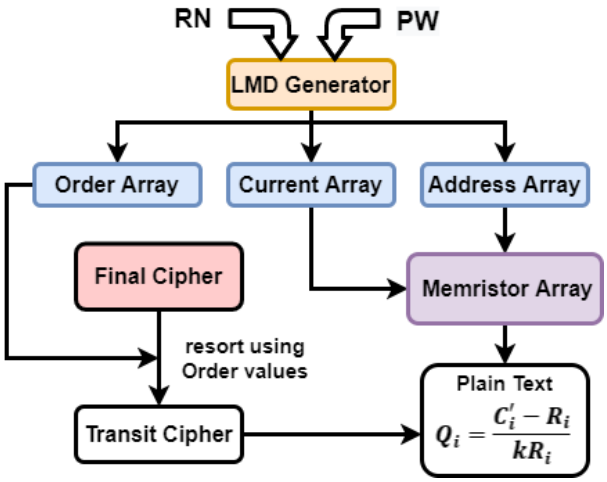


Fig. 3: Flowchart detailing the decryption process using the memristor PUF [20].

combined with a resistance value extracted from the original response at a particular cell when a certain current is injected. The address and current values are extracted from the LMDs. The order array is used to shuffle the cipher obtained from different symbols, in order to add another layer of confusion for the hacker [1], [20]. The memristor PUF which have been used for implementing the protocol are the Re-RAM PUF.

Figure 3 shows the extraction of plain text from the cipher in the decryption phase utilizing the handshake between the sender and receiver. The receiver can have the same blocks for addresses, orders, and currents, thereby extracting the resistances at the same cells addresses when the same current is injected as the encryption phase. These resistances will be used to retrieve the blocks of plain text from the cipher. However, according to [1], the challenges that this protocol faces using ReRAM PUFs are the cell-to-cell and array-to-array variations of the resistance, and current values due to the changes in the temperature other environmental factors.

Under different temperatures, the same cells operating under the same currents will generate noisy resistances. Having any error in reading the memristors causes failing to retrieve the blocks of plaintext. To avoid this, we propose using ECCs such as Reed Solomon (RS) or BCH codes in this protocol.

C. Error Correction Codes (ECC)

A variety of options are available today in the communication industry to detect and correct different errors that may appear while transmitting a message depending on their need. ECC are traditionally used to correct a received message which travels through a communication channel. For this purpose, parity bits are added as extra information to the message bits and then transmitted along the communication channel. With the advancement of time, many ECC have been introduced which are able to correct long messages with greater efficiency. These schemes are generally complex end require more power and advanced hardware. These mechanisms are also efficient when they are utilized with a larger number of message bits. When used for short blocks of message they tend to not deliver desirable results. Therefore, we propose to use simpler ECC mechanisms such as Reed Solomon and BCH codes for our protocol. These simple mechanisms are easier to implement on the hardware and will require low power.

D. Reed Solomon (RS) Error Correction

Reed Solomon codes are a group of ECC that operate on a block of data treated as a set of finite field elements called symbols. Therefore, RS codes are able to detect and correct multiple symbol errors. By adding t check symbols to the data, a RS code can detect (but not correct) any combination of up to and including t erroneous symbols, or locate and correct up to and including $\lfloor \frac{t}{2} \rfloor$ erroneous symbols at unknown locations. As an erasure code, it can correct up to and including t erasures at locations that are known and provided to the algorithm, or it can detect and correct combinations of errors and erasures. RS codes are also suitable as multiple-burst bit-ECC, since a sequence of $b+1$ consecutive bit errors can affect at most two symbols of size b. The choice of t is up to the designer of the code and may be selected within wide limits. There are two basic types of RS codes, original view and Bose Chaudhuri Hocquenghem codes (BCH) view, with BCH view being the most common as BCH view decoders are faster and require less working storage than original view decoders. The BCH codes form a class of cyclic ECC that are constructed using polynomials over a Galois field. Cyclic ECC means that a block code, where the circular shifts of each codeword gives another word that belongs to the code [21]–[24].

III. Proposed Protocol

In this work, we propose to implement known ECCs to correct the errors induced during the decryption phase when a noisy response of the memristor PUF is used. The message or plain text is encrypted using its ASCII equivalent divided into 4-bit symbols Q_i. Initially, these values were directly used in equation (1) to extract a cipher text.

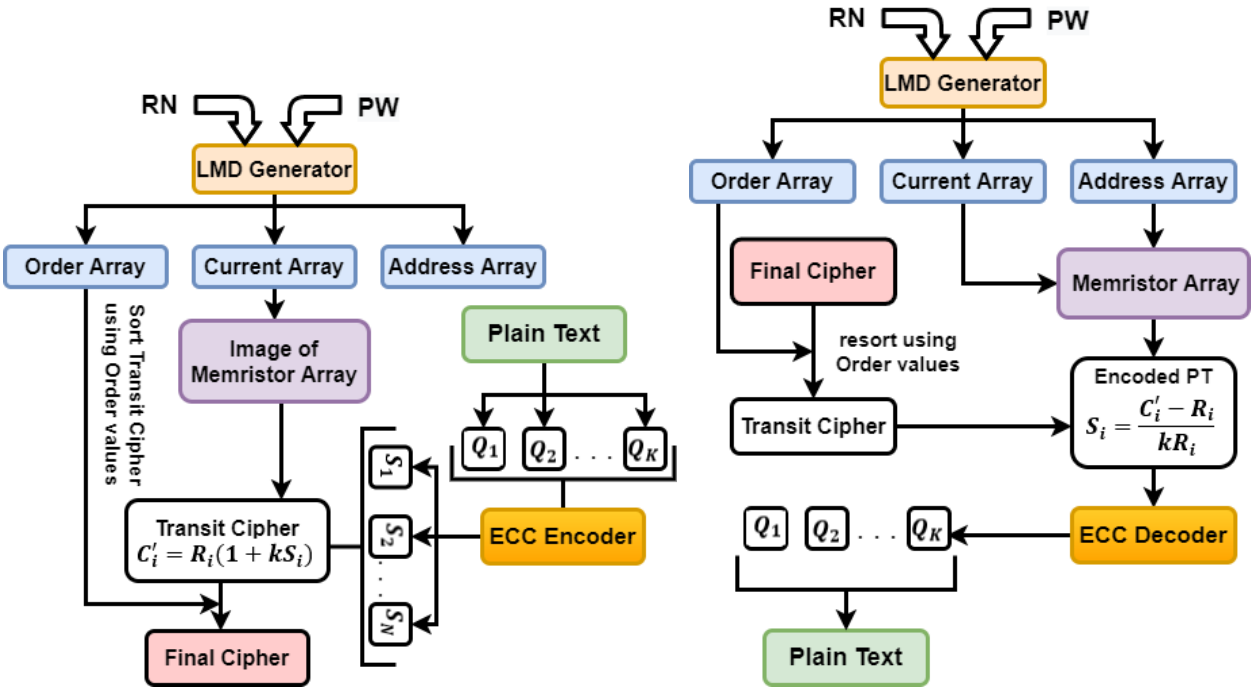
$$\boldsymbol{C_i = R_i(1 + KQ_i)} \tag{1}$$

As discussed before, the PUF responses are noisy, thereby leading to a noisy decrypted message. We will utilize ECCs to detect and correct the erroneous message. The detailed description of the protocol is presented below:

A. Detailed description of the Improved keyless encryption with ECC

Using the concepts of APG, we initially calculate the MD and LMD as shown in Figure 1. The number of bits required for each of these arrays depends on the number of ReRAM cells and the number of currents these cells are measured.

The PT is first divided into 4-bit symbols Q_i from its ASCII equivalent. These message symbols can be converted into the Gray code format before encoding them using the ECC. The message symbols are encrypted using block encoding techniques of the ECC encoder to add parity message bits to each message block (Collection of symbols). The ECC block will allow the extraction of the message when noise is introduced in the decryption phase of the protocol. This



(a) Encryption protocol.

(b) Decryption protocol.

Fig. 4: Improved Keyless Protocol with ECC.

encoded message S_i acts like the plaintext and is encrypted using equation 2

$$C_i = R_i(1 + KS_i) \tag{2}$$

Figure 4a shows the encapsulation of the message using the PUF in order to generate a cipher text. Here, S_i refers to each encoded message symbol whereas R_i refers to the resistance extracted from a cell address (A_i) at a current (Cr_i), C_i is the Transit Cipher and K is a constant. After extracting Transit Ciphers from each message block, they are reordered based on the order of the block in an ascending order and sent to the client as Final Ciphers C_i.

Figure 4b shows the decryption phase of the message from cipher by the client device. As the password and random number are shared information and the client has access to the PUF device, the LMD is obtained. The address, current and order arrays are derived from the LMD. The Final Ciphers C_i is de-shuffled to obtain back the transit cipher C_i' using the order array.

The resistance value R_i from the PUF device is extracted at a certain cell address (A_i) when a particular current (Cr_i) is applied for the ReRAM cell "i". These extracted resistance values are the noisier versions of the resistances extracted on the server end from where the message was transmitted. Using the decryption formula $S_i = \frac{C_i' - R_i}{KR_i}$, the encoded message S_i will be extracted.

The extracted encoded block is a noisy version of the encoded message as it is similar to a message sent through a communication channel. This noisy encoded block S_i is applied to an ECC decoder to extract Q_i which is used to obtain PT.

IV. Implementation and Results

The protocol has been implemented using real data set of resistances measured from 128 cells of a ReRAM array at 8 different currents $(100nA, 200nA \ldots 800nA)$, which are augmented with induced Additive White Gaussian Noise (AWGN) with different power resulting in different Signal to Noise Ratios (SNRs) to simulate different responses of ReRAM PUF under various environmental conditions. This experiment was repeated 1000 times and the noise was randomly selected each time based on the SNR.

We follow the proposed protocol described in Section III to encrypt our messages. The MD is sent to a circular shift mechanism used as eXtended Output Function (XOF) in this protocol and is employed on the MD n times and the circular shifted output is hashed with an SHA-512 to extract $512 * n = N$ bits of the LMD. Once we extract an LMD, we divided it into current, address and order arrays of 3, 7, and 6 bits, respectively. We used 3 bits to address the 8 different current values and 7 bits to address the 128 different cells. To allow an equal number of bits for every block without padding, we used 6 bits to get an order number to shuffle the bits.

We initially utilized RS codes for our testing. We encoded a 21 symbol message using block coding techniques and added parity symbols for every block of symbols. We used a RS (15, 7), which has the capability to correct 4 symbols as our

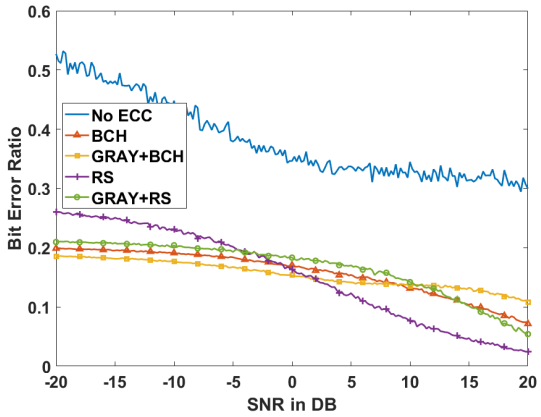


Fig. 5: Comparison of Bit Error Ratio between improved protocol with BCH and RS and original protocol.

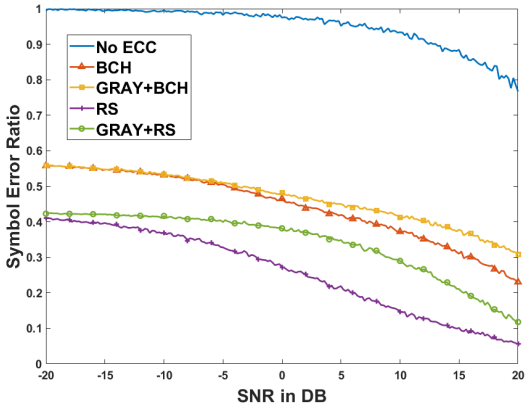


Fig. 6: Comparison of Symbol Error Ratio between improved protocol with BCH and RS and original protocol.

ECC block. Therefore, for every 7 message symbols Q_i in a block, we extracted 15 symbols of encoded message block S_i. Thereby, totaling 45 S_i's for 21 Q_i. These 45 S_i were then encrypted using equation (2). The cipher consisting of 45 C_i is sent to the receiver. On the receiver end, we use the 45 C_i in the Keyless decryption protocol. The decrypted 45 C_i will give us 45 S_i. These decrypted S_i is sent to RS (15, 7) decoder, where each 15 Q_i symbols will extract 7 message symbols, thereby extracting 21 Q_i in total from the 45 S_i.

The RS codes were able to correct all the message errors when they were in the correction capability of the RS codes. When the number of errors is higher than the codes capability, the Symbol Error Ratio (SER) follows the trend where the SER decreases with increase in SNR (Noise injected decreases as SNR increases). With the inclusion of Gray codes, we expected a considerable improvement in the Bit Error rate, but there was very subtle change in the result.

We also implemented the same protocol using BCH (63bits/ 16 Symbols, 30 bits/ 7 symbols) code to test the performance of BCH in a Keyless Protocol setting. We had to pad the message bits in order to match the BCH message length logistics. The comparison between RS and BCH in conjunction with Keyless protocol proposed in [1] is shown in Figures 5 and 6.

V. Conclusion and Future Work

The improved protocol for Keyless Encryption using ECC has allowed decryption of the messages from a Noisy PUF response. The noisy PUF response is analogous to noise injected through the communication channel, which in this context is generated due to the impact of aging and environmental changes on the PUF behavior. The results show that, without the inclusion of ECC block, the decrypted message contains errors. The proposed protocol has allowed proper decryption of the message when the noise is in the range of its correction capability. Even when the noise injected is more than the ECC block can correct, the resulting message contains less errors than the one resulting from the noisy PUF. This improved protocol with ECC will allow the IoT devices to use the keyless encryption protocol in order to allow practical implementation of this protocol. Keyless encryption protocol proposed does not come with too much complexity overhead, thereby allowing low power devices to use it in encrypting their sensitive information shared.

References

[1] B. Cambou, D. Hély, and S. Assiri, "Cryptography with analog scheme using memristors," J. Emerg. Technol. Comput. Syst., vol. 16, no. 4, Sep. 2020. [Online]. Available: https://doi.org/10.1145/3412439

[2] P. Kocher, J. Jaffe, B. Jun et al., "Introduction to differential power analysis and related attacks," 1998.

[3] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in Annual international cryptology conference. Springer, 1999, pp. 388–397.

[4] P.Kocher, J. B.Jun, and al.et, "Introduction to differential power analysis," Journal of Cryptographic Engineering, vol. 1, pp. 5–27, 2011.

[5] N. Baracaldo, L. A. Bathen, R. Ozugha, R. Engel, S. Tata, and H. Ludwig, "Securing data provenance in internet of things (iot) systems," in Service-Oriented Computing – ICSOC 2016 Workshops, K. Drira, H. Wang, Q. Yu, Y. Wang, Y. Yan, F. Charoy, J. Mendling, M. Mohamed, Z. Wang, and S. Bhiri, Eds. Cham: Springer International Publishing, 2017, pp. 92–98.

[6] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," Future generation computer systems, vol. 29, no. 7, pp. 1645–1660, 2013.

[7] C. Herder, M. Yu, F. Koushanfar, and S. Devadas, "Physical unclonable functions and applications: A tutorial," Proceedings of the IEEE, vol. 102, no. 8, pp. 1126–1141, 2014.

[8] M. Yu and S. Devadas, "Secure and robust error correction for physical unclonable functions," IEEE Design Test of Computers, vol. 27, no. 1, pp. 48–65, 2010.

[9] R. Maes, Physically Unclonable Functions: Constructions, Properties and Applications, 1st ed. Springer Publishing Company, Incorporated, 2016.

[10] A. R. Korenda, F. Afghah, and B. Cambou, "A secret key generation scheme for internet of things using ternary-states reram-based physical unclonable functions," in 2018 14th International Wireless Communications Mobile Computing Conference (IWCMC), 2018, pp. 1261–1266.

[11] A. R. Korenda, F. Afghah, B. Cambou, and C. Philabaum, "A proof of concept sram-based physically unclonable function (puf) key generation mechanism for iot devices," in 2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), 2019, pp. 1–8.
[12] M. Hiller, L. Kürzinger, and G. Sigl, "Review of error correction for pufs and evaluation on state-of-the-art fpgas," Journal of Cryptographic Engineering, vol. 10, pp. 229–247, 9 2020. [Online]. Available: https://doi.org/10.1007/s13389-020-00223-w
[13] D. Merli, F. Stumpf, and G. Sigl, "Protecting puf error correction by codeword masking." IACR Cryptol. ePrint Arch., vol. 2013, p. 334, 2013.
[14] A. Shamsoshoara, A. Korenda, F. Afghah, and S. Zeadally, "A survey on physical unclonable function (puf)-based security solutions for internet of things," Computer Networks, vol. 183, p. 107593, 2020.
[15] B. Cambou, "Puf-based password generation scheme," 2019, uS Patent US20180129801A1.
[16] L. Chua, "Memristor-the missing circuit element," IEEE Transactions on Circuit Theory, vol. 18, no. 5, pp. 507–519, 1971.
[17] B. F. Cambou, R. C. Quispe, and B. Babib, "Puf with dissolvable conductive paths," May 28 2020, uS Patent App. 16/493,263.
[18] J. Kim, H. Nili, G. C. Adam, N. D. Truong, D. B. Strukov, and O. Kavehei, "Predictive analysis of 3d reram-based puf for securing the internet of things," in 2018 IEEE Region Ten Symposium (Tensymp). IEEE, 2018, pp. 91–94.
[19] B. F. Cambou, "Password management with addressable physical unclonable function generators," 2019, uS Patent App. 16/415,235.
[20] Y. Zhu, B. Cambou, D. Hely, and S. Assiri, "Extended protocol using keyless encryption based on memristors," in Science and Information Conference. Springer, 2020, pp. 494–510.
[21] W. A. Geisel, Tutorial on Reed–Solomon Error Correction Coding. NASA, 1990.
[22] K. A. S. Immink, Reed–Solomon Codes and Their Applications. IEEE Press, 1994.
[23] J. Hong and M. Vetterli, "Simple algorithms for bch decoding," IEEE Transactions on Communications, vol. 43, pp. 2324–2333, 1995.
[24] R. Koetter and A. Vardy, Algebraic soft-decision decoding of Reed-Solomon codes. IEEE Transactions on Information Theory, 2003, vol. 49, no. 11.