Cryptanalysis of two recently proposed PUF based authentication protocols for IoT: PHEMAP and Salted PHEMAP

Morteza Adeli[1], Nasour Bagheri[2]

[1] Department of Science, Shahid Rajaee Teacher Training University, Tehran, Iran, Postal code: 16788-15811, Tel/fax:+98-21-22970117, M.adeli@sru.ac.ir
[2] Electrical Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Iran, NBagheri@srttu.edu

Abstract. Internet of Things(IoT) consists of a large number of interconnected coexist heterogeneous entities, including Radio-frequency identification(RFIDs) based devices and other sensors to detect and transfer various information such as temperature, personal health data, brightness, etc. Security, in particular, authentication, is one of the most important parts of information security infrastructure in IoT systems. Given that an IoT system has many resource-constrained devices, a goal could be designing a proper authentication protocol that is lightweight and can resist against various common attacks, targeting such devices. Recently, using Physical Unclonable Functions (PUF) to design lightweight authentication protocols has received a lot of attention among researchers. In this paper, we analyze two recently proposed authentication protocols based on PUF chains called PHEMAP and Salted PHEMAP. We show that these protocols are vulnerable to impersonate, desynchronization and traceability attacks.

keywords IoT; authentication; PUF; security analysis.

1 Introduction

Internet of Things(IoT) is growing rapidly nowadays and researchers are studying and developing different aspects of IoT applications. For example, IoT could be used in smart homes, where IoT devices such as sensors are used to control temperature, light and house security to improve the quality of life. An IoT network has several layers, from edge devices to cloud-based servers. The edge devices include a variety of coexisting devices that could expand from very constraint devices, e.g. RFID passive tags, to general superpose computers with reasonable resources. Among them, RFID has fundamental importance in IoT, thanks to its cost-efficiency. An RFID based sensor can work in various environments without significant artificial interference, with low energy consumption, to detect, store and send information through wireless channels.

Generally, in RFID systems, a unique identity(ID) is assigned to each tag to find and recognize a specific device. When a reader wants to receive desired data from a tag, the reader lunches an authentication process to identify the target tag. So far, many authentication protocols for different applications and environments have been proposed in the literature. Since the tags usually have restricted computation power and storage size, they support only lightweight operations such as exclusive OR (XOR), pseudo-random number generator (PRNG), shift operation, etc. Today, using physical unclonable function (PUF), in authentication protocols have been studied by many researchers and several PUF based authentication protocols have been proposed in the literature [9,1,21,13,12,11]. A PUF works as a digital fingerprint and serves as a unique identity for a device [19]. When a device (like FPGA) is fabricated in the manufactory, a PUF entity is embodied in the physical structure which is unique and infeasible to duplicate or predict. An ideal PUF is expected to operate as a one-way function so we can use it in protocols that are based on challenge-response pairs as a security component. PUF architectures for silicon devices are mainly classified in two classes [17]: (1) delay-based such as the arbiter PUF, the ring oscillator (RO) PUF, and the Anderson PUF, that use differences in paths delays within the specific circuit (2) memory-based such as the SRAM PUF, butterfly PUF, sense amplifier PUF, flip-flop PUF, that exploit the mismatches of memory cell to generate a response to a challenge.

In this paper, we do not focus on how to design an efficient PUF, therefore we suppose that the PUFs used in the authentication schemes have good behavior, for example, enough stable and unpredictable. It worth noting, to study the security of cryptographic construction, it is common to use idealized versions of some crypto-primitives, e.g. pseudo-random functions (PRFs) or pseudorandom generators (PRGs). An ideal PUF plays a similar role in the case of PUF-based authentication protocols. Although no one has ever built an ideal PUF, intensive research is being done to build PUFs with better properties, such as good entropy and small or even zero bit error rates. This approach is compliant with previous researches that are using PUF to design authentication protocols for IoT systems, e.g. [2,15,7,20,14].

1.1 Related works

Majzoobi *et al.* [18] introduced a Slender PUF protocol and claimed to be efficient and secure. However, later analysis demonstrated its pitfalls such as the lack of privacy [3,10]. Aysu *et al.* [3] presented an efficiently PUF based mutual authentication scheme between a server and a resource-constrained device. In their report, they showed how each component of the proposed scheme can be implemented efficiently on a resource-constrained platform such as SASEBO-GII board.

Kulseng *et al.* [16] proposed a mutual authentication and ownership transfer protocol based on PUF and Linear Feedback Shift Registers (LFSR). They claimed that their scheme can be implemented efficiently on hardware and is resistant against various attacks. Unfortunately, Xu *et al.* [20] showed that their

claim is wrong and presented a desynchronization attack on it. Then, they proposed a lightweight authentication protocol based on PUF. Bendavid *et al.* [6] looked closely at Xu *et al.* scheme and showed that their scheme is vulnerable to desynchronization attack and secret disclosure attack. Braeken [7], first showed that the PUF based key agreement scheme, presented by Chatterjee *et al.* [8] is vulnerable to impersonation, replay, and man-in-the-middle attacks and next, to address this weakness, they proposed a new efficient key agreement scheme based on the PUF. In 2019, Ameri *et al.* [2] proposed two PUF based authentication schemes for high-resource and low-resource devices and proved that their scheme can resist against known attacks. Gope *et al.* [11] proposed a PUF-based mutual authentication protocol for real-time data access in Industrial Wireless Sensor Networks (IWSN).

One of the challenges to using most of the above PUF based authentication protocols is the very large number of challenge-response pairs of PUF need to be stored by the authenticator and the devices embedding the PUF. On the other hand, in real applications, typically many devices of an IoT system are resource-constrained. Hence, most of the existing PUF based mutual authentication protocols are impractical and only able to verify the identity of the devices. One solution to reduce the number of challenge-response pairs is to construct sequences (chains) of challenge-response pairs by a recursive invocation of the PUF embedded on the devices. In this direction, recently Barbareschi *et al.* [4,5] proposed two mutual authentication protocols (called PHEMAP and Salted PHEMAP scheme) for low-cost hardware which use PUF chains in their authentication procedures. The Salted PHEMAP has been specially designed for cloud-edge(CE) IoT systems. They analyzed their schemes through formal and informal security proof and claimed that their schemes are secure against known attacks. In this paper, we analyze the security of these protocols in more detail and provide the first third-party security analysis of them, to the best of our knowledge. More precisely, first, we analyze the PUF based authentication protocol called PHEMAP, proposed by Barbareschi *et al.* [4] and show that this scheme is vulnerable to impersonate attack. Furthermore, we demonstrate that the PHEMAP scheme is traceable. In the following, we analyze the Salted PHEMAP protocol proposed by Barbareschi *et al.* [5] for Cloud-Edge(CE) IoT systems. This scheme is also vulnerable to impersonate, desynchronization and traceability attacks.

1.2 Organization

The remainder of this paper is organized as follows: in section section 2 the required preliminaries are provided, including a brief description of the PHEMAP scheme [4] and Salted PHEMAP scheme [5], that has been designed for cloud-edge(CE) IoT systems. In section 3 we explain how to impersonate and traceability attacks can be performed on the PHEMAP scheme and also we will explain the weakness of the Salted PHEMAP scheme. In the end, the conclusion of the paper is described in section 4.

2 Preliminaries

Through the paper we are using the notation represented in Table 1.

Table 1. Notation used in this paper

Notation	Description
$\gamma_{D,c_0,M}$	a PUF chain of device D with root chain c_0 and length M
$\theta(.)$	Physical unclonable function
l_i	i-*th* link in chain γ
n	nonce generated in the verifier
r	random number generated in the tag
S	sentinel period

2.1 PHEMAP scheme

In this section, we give a brief description of the PHEMAP scheme [4]. This scheme uses only PUF and XOR functions to encrypt and decrypt messages transferred between a verifier and a tag. The proposed scheme contains three phases as following: (1) enrollment and (2) initialization and (3) verification. Before we describe the PHEMAP scheme, we need some definitions, that are taken from [4].

Definition: Let $\theta_D(.)$ be PUF inserted in device D. The PUF chain $\gamma_{D,c_0,M}$ with root chain c_0 and length M is defined as:

$$\{c_0, \theta_D(c_0), \theta_D^2(c_0), ..., \theta_D^{M-1}(c_0)\} \tag{1}$$

where $\theta_D^i(.) = \overbrace{\theta_D(\theta_D(...))}^{i}$ and all of $\theta_D^i(.)$ are distinct. We referred each $\theta_D^i(c_0)$ to as *links* and noted by l_i hereafter.

Definition: Let $\gamma_{D,c_0,M}$ be a chain, σ_0 be a link on it and S be a positive integer. We refer to as chain *sentinels* all multiple of S, starting from link σ_0.

Enrollment In this phase, the verifier generates T distinct chains $\gamma_{D,c_0,M}$ where each root chain c_0 is selected randomly, so each link appears only once over the extracted chains. The length of each chain (M) is different and depends on the number of the new distinct links that can be generated by iterating the PUF starting from the random root chain c_0. All of T generated chains are stored in the verifier and the devices need to store only the last synchronized link that has been used in previous exchanges. The number of generated chains depends on the storage capacity of the verifier and the number of devices that can be managed by it. In the end, the *sentinel* period S, is defined and embedded in both the devices and the verifier.

Initialization The initialization contains four phases as following:

1. The verifier generates a random nonce n and sends

$$msg1 = \{l_i, (\oplus_{j=0}^{S-3} l_{i+j+1}) \oplus n, l_{i+S-1} \oplus n\} = \{l_i, v_1, v_2\} \tag{2}$$

   to the device D.
2. Upon receiving the message *msg1*, the device D checks

$$\oplus_{j=0}^{S-2} \theta_D^{j+1}(l_i) \stackrel{?}{=} v_1 \oplus v_2 \tag{3}$$

   If it holds true, the device D generates a random nonce r and computes

$$msg2 = \{\theta_D^S(l_i) \oplus r, \theta_D^{S+1}(l_i) \oplus r\} = \{d_1, d_2\} \tag{4}$$

   and sends it to the verifier. Moreover, the device D saves d_2 in its secure register.
3. The verifier computes and checks

$$l_{i+S} \oplus l_{i+S+1} \stackrel{?}{=} d_1 \oplus d_2 \tag{5}$$

   If it's be true, the verifier authenticates the device D and sends

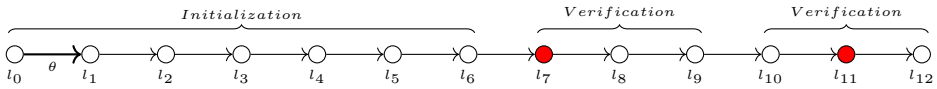
$$msg3 = \{l_i, l_{i+S+2} \oplus r\} = \{l_i, v_3\} \tag{6}$$

   to D.
4. D computes and checks

$$\theta_D^{S+1}(l_i) \oplus \theta_D^{S+2}(l_i) \stackrel{?}{=} v_3 \oplus d_2 \tag{7}$$

   If it holds true, the device D authenticates the verifier and saves $\theta_D^{S+2}(l_i)$ in its register.

Verification Both of the verifier and the device D can be initiated this phase. Suppose that the verifier is initiator of the protocol. The verifier knows the last synchronized link l_i and the first sentinel link σ_0 after initialization. If $l_{i+1} \neq \sigma_0$, the verifier sends l_{i+1} to the device D, otherwise, it sends l_{i+2} to it. The device D has the last synchronized link l_i in its register and knows the sentinel link σ_0 based on the current value of its counter. So it computes $l'_{i+1} = \theta(l_i)$ and checks $l_{i+1} \stackrel{?}{=} l'_{i+1}$(or if $l_{i+1} = \sigma_0$, it checks $l_{i+2} \stackrel{?}{=} l'_{i+2}$). If it holds, the device D authenticates the verifier and saved l_{i+1}(or l_{i+2}) in its register. Next, the device D runs the same approach to authenticated by the verifier. In Figure 1, the PHEMAP scheme has been illustrated by an example. In this example, first link is l_0 and sentinel links are l_7, l_{11} and $S = 4$.



Phase	Verifier		Device
Init.	$v_1 = l_1 \oplus l_2 \oplus n$ $v_2 = l_3 \oplus n$	$\xrightarrow{msg1=\{l_0, v_1, v_2\}}$	
		$\xleftarrow{msg2=\{d_1, d_2\}}$	$\theta_D(l_0) \oplus \theta_D^2(l_0) \oplus \theta_D^3(l_0) \stackrel{?}{=} v_1 \oplus v_2$ $d_1 = \theta_D^4(l_0) \oplus r$ $d_2 = \theta_D^5(l_0) \oplus r$ $storeQ = d_2$
	$l_4 \oplus l_5 \stackrel{?}{=} d_1 \oplus d_2$ $v_3 = l_6 \oplus r$	$\xrightarrow{msg3=\{l_0, v_3\}}$	
			$\theta_D^5(l_0) \oplus \theta_D^6(l_0) \stackrel{?}{=} v_3 \oplus d_2$ $storeQ = \theta_D^6(l_0)$
Verif.	$l_0' = l_8$ $l_1' = l_9$	$\xrightarrow{l_8}$	
		$\xleftarrow{\theta_D(l_8)}$	$\theta_D^2(Q) \stackrel{?}{=} l_8$ $storeQ = \theta_D(l_8)$
	$l_1' \stackrel{?}{=} \theta_D(l_8)$		

Fig. 1. PHEMAP protocol, where Init. and Verif. denote initialization and verification retrospectively

2.2 Salted PHEMAP scheme

In this section, we give briefly description of Salted PHEMAP scheme, proposed by Barbareschi *et al.* [5] to be implemented in Cloud-Edge(CE) based IoT systems. CE-based IoT systems are typically consist of three architectural layers, see Figure 2: i) cloud service as top layer ii) gateway nodes as middle layer iii) terminal nodes or edge IoT devices as lower layer.

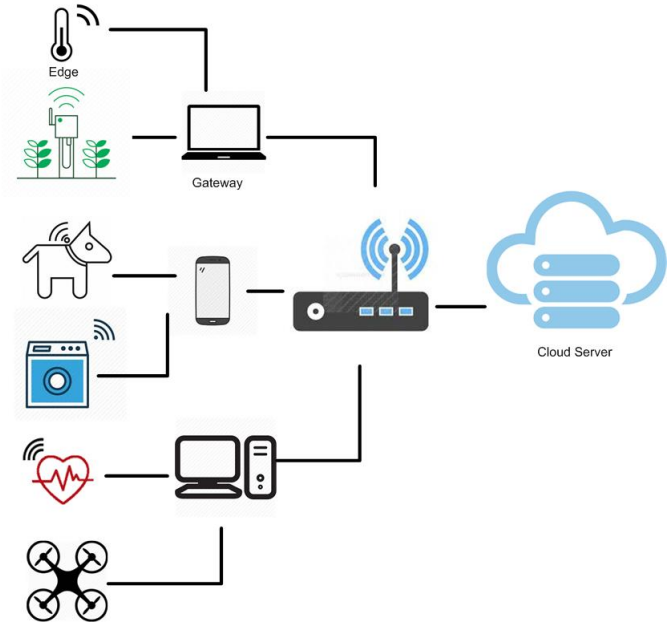


Fig. 2. CE-based architecture

The salted PHEMAP scheme provides mutual authentication between a terminal node and the respective gateway in a CE-based IoT system. In salted PHEMAP, gateways act as a local verifier for underlying terminal nodes by part of enrolled PUF chains that get from authentication service(AS). Note that the Salted PHEMAP scheme is always performed after to the basic PHEMAP scheme and also suppose that the terminal device D and AS have been synchronized on link l_{i-1} of chain $\gamma_{D,l_0,M}$. The Salted PHEMAP scheme starts with the setup phase as follows:

1. Device D sends to AS a request message $m_0 = \{\theta(l_{i-1})\}$ and also saves l_i in its local memory.
2. AS verifies that received message m_0 is equal to immediately following link l_i that is in the chain $\gamma_{D,l_0,M}$. If it holds true, AS extract a carnet $\tau_{D,t_0,T} = \{t_0, t_1, ..., t_{T-1}\}$ from chain $\gamma_{D,l_0,M}$. Next AS generates random salt S_D and computes message $m_1 = \{v_1, v_2\} = \{\theta(l_1), \theta^2(l_1) \oplus S_D\}$ to device D.
3. Device D computes $l_2 = \theta(l_1)$ and compares it with v_1. If the two values match, device D authenticates the AS and extracts salt S_D from v_2. Next it computes $l_3 = \theta(l_2)$ and sends message $m_2 = l_3$ to the AS. It saves $\theta(l_3) \oplus S_D$ in its local memory to communicate with respective gateway.
4. AS checks $\theta(l_2) \stackrel{?}{=} m_2$ and if the two values equal, then AS computes a salted carnet $\chi_{D,x_0,T} = \{x_0, x_1, ..., x_{T-1}\} = \{t_0 \oplus S_D, t_1 \oplus S_D, ..., t_{T-1} \oplus S_D\}$ and sends message $m_3 = \{\chi_{D,x_0,T}\}$ to gateway through a secure channel.

Now, the device D and the respective gateway G use $\{\chi_{D,x_0,T}\}$ to authentication operation between themselves. Authentication operations between the device D and the gateway G is look like to operations between AS and the device D in the basic PHEMAP scheme. We illustrate the Salted PHEMAP scheme by an example in Figure 3. In this example, suppose that l_0 is synchronized link between AS and D.

Phase	**Authentication Service**	**Gateway**	**Device**
Init.		$\xleftarrow{m_0=\{l_1\}}$	compute $l_1 = \theta(l_0)$
	$l_1 \stackrel{?}{=} m_0$ $\tau_{D,t_0,T} = \{t_0, t_1, ..., t_{T-1}\}$ $S_D : random$ $v_1 = l_2$ $v_3 = l_3 \oplus S_D$		
		$\xrightarrow{m_1=\{v_1,v_2\}}$	$\theta(l_1) \stackrel{?}{=} v_1$ $S_D = \theta^2(l_1) \oplus v_2$ store $\theta^4(l_1) \oplus S_D = l_5 \oplus S_D$
	$l_4 \stackrel{?}{=} \theta^3(l_1)$ $\chi_{D,t_0,T} = \{x_i = t_i \oplus S_D, t_i \in \tau_{D,t_0,T}\}$	$\xleftarrow{m_2=\{\theta^3(l_1)\}}$	
	$\xrightarrow[to\ gateway]{m_3=\{\chi_{D,t_0,T}\}}$		
Verif.		$L_1 = x_1$ $L_2 = x_2$ $\xrightarrow{L_1}$	
			$\theta(x_0) \stackrel{?}{=} L_1$ $storeQ = \theta^2(x_0)$ $\xleftarrow{\theta^2(x_0)}$
		$L_2 \stackrel{?}{=} \theta^2(x_0)$	

Fig. 3. Salted PHEMAP protocol, where Init. and Verif. denote initialization and verification retrospectively

3 Security analysis of PHEMAP and Salted-PHEMAP protocol

3.1 Security challenges of the PHEMAP protocol

Impersonate attack In the initialization phase of the PHEMAP protocol, the attacker intercepts three valid data transmitted between the verifier and the device:

- The query message of the verifier $msg1 = \{l_0, v_1, v_2\}$
- The response message of the device $msg2 = \{d_1, d_2\}$
- The response message of the verifier $msg3 = \{l_0, v_3\}$

Next, the attacker repeats message *msg1* and intercepts response message of the device $msg'2$. Next, the attacker computes

- $\triangle r = d_1 \oplus d'_1 = r \oplus r'$.
- $v'_3 = v_3 \oplus \triangle r = l_6 \oplus r'$

and sends $msg'3 = \{l_0, v'_3\}$ to the device. Upon receiving the message $msg'3$, the device authenticates the attacker as a legitimate verifier.

Traceability In the initialization phase, the attacker intercepts the query message $msg1$ sent by verifier and the response message of the device $msg2 = \{d_1, d_2\}$. We know that summation of d_1 and d_2 is a fix value, because

$$d_1 \oplus d_2 = (\theta_D^4(l_0) \oplus r) \oplus (\theta_D^5(l_0) \oplus r) = (\theta_D^4(l_0) \oplus \theta_D^5(l_0)) \tag{8}$$

So the attacker can trace a target device D by sending the fixed message $msg1$ to it and computing the sum of the response message $msg2$ of the device D.

3.2 Security challenge of the Salted PHEMAP protocol

Desynchronization attack In desynchronization attack on salted PHEMAP, the attacker attempts to restrict access of a legitimate device to its respective gateway. In setup phase, the attacker intercepts message $m_1 = \{v_1, v_2\} = \{\theta(l_i), \theta^2(l_i) \oplus S_D\}$ and modifies it to $m'_1 = \{\theta(l_i), \theta^2(l_i) \oplus S_D \oplus \triangle\}$ and sends its to the device D. The device D doesn't verify the integrity of the message m'_1, so it computes $S_D = \theta^2(l_i) \oplus v_2 \oplus \triangle$. Therefore the device D, unlike the gateway G, is synchronized to link $\theta^4(l_i) \oplus S_D \oplus \triangle$ and so, the verification operation between the device D and the gateway G will be failed.

Impersonation attack Suppose that the basic PHEMAP and subsequently, the salted PHEMAP authentication scheme has been run and all of the transferred messages between a device D and the authentication service AS has been intercepted by the attacker. Let the initiate link of the PUF chain be l_0. If the attacker sends the query message $\{l_0, v_1, v_2\}$ to the device D, it goes to the initialization phase of the basic PHEMAP authentication scheme. According to subsection 3.1, the attacker impersonates a legitimate authentication service AS. Therefore, the attacker impersonates himself as the authentication service AS and the gateway G.

Traceability attack In setup phase, three links l_1, l_2, l_4 have fixed values. So if the attacker intercepts the message m_0, he can trace the target device D by resending the message m_0 and tracing the response messages m_2, m_3.

4 Conclusion

In this paper, we analyzed the security of two recently proposed PUF-based protocols, i.e. PHEMAP and Salted PHEMAP. The detailed analysis has shown that these protocols are vulnerable to attacks such as device impersonation and traceability attacks.

References

1. M. N. Aman, K. C. Chua, and B. Sikdar. Mutual authentication in IoT systems using physical unclonable functions. *IEEE Internet of Things Journal*, 4(5):1327–1340, Oct 2017.
2. M. H. Ameri, M. Delavar, and J. Mohajeri. Provably secure and efficient PUF-based broadcast authentication schemes for smart grid applications. *International Journal of Communication Systems*, 32(8):e3935, 2019. e3935 dac.3935.
3. A. Aysu, E. Gulcan, D. Moriyama, P. Schaumont, and M. Yung. End-to-end design of a puf-based privacy preserving authentication protocol. In T. Güneysu and H. Handschuh, editors, *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*, volume 9293 of *Lecture Notes in Computer Science*, pages 556–576. Springer, 2015.
4. M. Barbareschi, A. D. Benedictis, and N. Mazzocca. A PUF-based hardware mutual authentication protocol. *Journal of Parallel and Distributed Computing*, 119:107 – 120, 2018.
5. M. Barbareschi, A. D. Benedictis, E. L. Montagna, A. Mazzeo, and N. Mazzocca. A PUF-based mutual authentication scheme for cloud-edges IoT systems. *Future Generation Computer Systems*, 101:246 – 261, 2019.
6. Y. Bendavid, N. Bagheri, M. Safkhani, and S. Rostampour. Iot device security: Challenging a lightweight RFID mutual authentication protocol based on physical unclonable function. *Sensors*, 18(12), 2018.
7. A. Braeken. Puf based authentication protocol for IoT. *Symmetry*, 10(8), 2018.
8. U. Chatterjee, R. S. Chakraborty, and D. Mukhopadhyay. A PUF-based secure communication protocol for IoT. *ACM Trans. Embed. Comput. Syst.*, 16(3):67:1–67:25, Apr. 2017.
9. W. Che, M. Martin, G. Pocklassery, V. K. Kajuluri, F. Saqib, and J. Plusquellic. A privacy-preserving, mutual PUF-based authentication protocol. *Cryptography*, 1(1), 2016.
10. J. Delvaux, R. Peeters, D. Gu, and I. Verbauwhede. A survey on lightweight entity authentication with strong PUFs. *ACM Comput. Surv.*, 48(2):26:1–26:42, 2015.
11. P. Gope, A. K. Das, N. Kumar, and Y. Cheng. Lightweight and physically secure anonymous mutual authentication protocol for real-time data access in industrial wireless sensor networks. *IEEE Trans. Industrial Informatics*, 15(9):4957–4968, 2019.

12. P. Gope, J. Lee, and T. Q. S. Quek. Lightweight and practical anonymous authentication protocol for RFID systems using physically unclonable functions. *IEEE Trans. Information Forensics and Security*, 13(11):2831–2843, 2018.
13. P. Gope and B. Sikdar. Lightweight and privacy-preserving two-factor authentication scheme for IoT devices. *IEEE Internet of Things Journal*, 6(1):580–589, Feb 2019.
14. C. Hristea and F. L. Tiplea. A PUF-based destructive private mutual authentication RFID protocol. In J. Lanet and C. Toma, editors, *Innovative Security Solutions for Information Technology and Communications - 11th International Conference, SecITC 2018, Bucharest, Romania, November 8-9, 2018, Revised Selected Papers*, volume 11359 of *Lecture Notes in Computer Science*, pages 331–343. Springer, 2018.
15. C. Hristea and F. L. Tiplea. Destructive privacy and mutual authentication in vaudenay's RFID model. *IACR Cryptology ePrint Archive*, 2019:73, 2019.
16. L. Kulseng, Z. Yu, Y. Wei, and Y. Guan. Lightweight mutual authentication and ownership transfer for RFID systems. In *2010 Proceedings IEEE INFOCOM*, pages 1–5, 2010.
17. R. Maes. *Physically Unclonable Functions: Constructions, Properties and Applications*. SpringerLink : Bücher. Springer Berlin Heidelberg, 2013.
18. M. Majzoobi, M. Rostami, F. Koushanfar, D. S. Wallach, and S. Devadas. Slender PUF protocol: A lightweight, robust, and secure authentication by substring matching. In *2012 IEEE Symposium on Security and Privacy Workshops, San Francisco, CA, USA, May 24-25, 2012*, pages 33–44, 2012.
19. K. Sha, W. Wei, T. A. Yang, Z. Wang, and W. Shi. On security challenges and open issues in internet of things. *Future Generation Computer Systems*, 83:326–337, 2018.
20. H. Xu, J. Ding, P. Li, F. Zhu, and R. Wang. A lightweight RFID mutual authentication protocol based on physical unclonable function. *Sensors*, 18(3), 2018.
21. J. Zhang, X. Tan, X. Wang, A. Yan, and Z. Qin. T2FA: Transparent two-factor authentication. *IEEE Access*, 6:32677–32686, 2018.