A Secure Lightweight RFID Mutual Authentication Protocol Without Explicit Challenge-Response Pairs

Keke Huang[1,2], Changlu Lin[1,2,3(✉)], and Yali Liu[2,4]

[1] College of Computer and Cyber Security, Fujian Normal University, Fuzhou 350117, Fujian, China
hkk@yjs.fjnu.edu.cn

[2] Fujian Provincial Key Lab of Network Security and Cryptology, Fuzhou 350007, Fujian, China

[3] School of Mathematics and Statistics, Fujian Normal University, Fuzhou 350117, Fujian, China
cllin@fjnu.edu.cn

[4] College of Computer Science and Technology, Jiangsu Normal University, Xuzhou 221116, Jiangsu, China
liuyali@jsnu.edu.cn

Abstract. Radio Frequency Identification (RFID) has been widely deployed to various scenarios, but its security and privacy issues need to be concerned due to the tag's limited computing and storage resources. While benefiting from the great convenience and advantages of RFID systems, security is still a considerable threat to their applications, such as desynchronization attacks and cloning attacks. In this paper, we propose a secure lightweight mutual authentication protocol based on the configurable tristate physical unclonable functions and the cryptographic primitive of verifiable secret sharing to solve these issues. More specifically, the tag equipped with the configurable tristate physical unclonable functions structure can enhance the tag's security and effectively resist machine learning modeling attacks. Verifiable secret sharing plays the role of decentralized storage of secrets, and ensures the verifiability of the correctness of the each shares. The verifiability provided by verifiable secret sharing is effective against tag impersonation attacks, and

Supported by the National Natural Science Foundation of China (U1705264, 61702237, 61872168); Natural Science Foundation of Fujian Province (2019J01275); Guangxi Key Laboratory of Trusted Software (KX202039); the Opening Foundation of Guangxi Key Laboratory of Cryptography and Information Security (Guilin University of Electronics Technology) (GCIS202114); the Ministry of Education University-Industry Collaborative Education Program of China (CXHZ-GWebRAY-202002-18); the Special Foundation of Promoting Science and Technology Innovation of Xuzhou City (KC18005); the Natural Science Foundation of Jiangsu Province (BK20150241); the Natural Science Foundation of the Higher Education Institutions of Jiangsu Province (14KJB520010); the Scientific Research Support Project for Teachers with Doctor's Degree of Jiangsu Normal University (14XLR035); the Jiangsu Provincial Government Scholarship for Overseas Studies.

J. Lin and Q. Tang (Eds.): AC3 2022, LNICST 448, pp. 79–107, 2022.
https://doi.org/10.1007/978-3-031-17081-2_6

the validity of the each share provided by the tag has to be verified before it is adopted. Finally, the correctness of our protocol is analyzed formally using BAN-logic and the its security is verified informally by the Scyther. In addition, we analyze security properties including data integrity, data confidentiality, anonymity, mutual authentication, forward security and resistance to various malicious attacks. The results show that the proposed protocol satisfies various security properties and resistance to diverse malicious attacks.

Keywords: Radio Frequency Identification (RFID) · Configurable Tristate PUF (CT PUF) · Verifiable Secret Sharing (VSS) · BAN logic · Scyther

1 Introduction

The main principle of Radio Frequency Identification (RFID) technology is to automatically identify and track objects, people, and other objects in an open environment through radio frequency signals [1]. RFID has the following advantages: the non-contact, the high reliability, the fast certification, the waterproof, the anti-magnetic, the heat-resistant, the long service life, the long contact distance and so on. With the development of the Internet of Things (IoT), RFID technology has been widely used in various industries and scenarios, such as supply chain management systems [2], the healthcare environment [3,4], the vehicle identification [5], the unmanned aerial vehicle (UAV) [6].

The structure of a typical RFID system [7] is depicted in Fig. 1. The back-end server, which is a powerful device that maintains a database with tag information, communicates with the reader via a secure channel, while the reader and the tag is a wireless communication channel that is vulnerable to adversary attacks, such as the desynchronization attacks [8], the replay attacks [2], the man-in-middle attacks [6], the physical and cloning attacks [9], etc. To address the critical challenges of low-cost RFID tag secure authentication, many lightweight authentication schemes based on different technologies and from different perspectives have been proposed. Some research works are reviewed as follows.

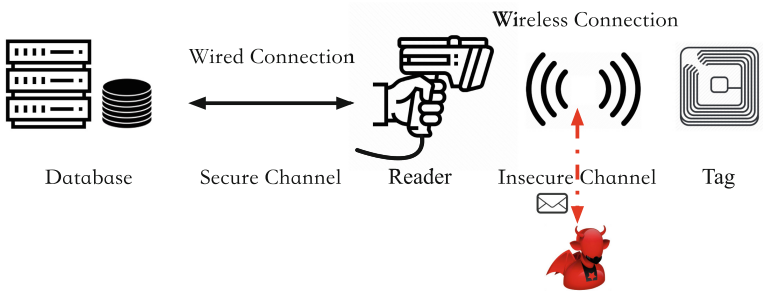

Fig. 1. The structure of a typical RFID system

Peris-Lopez et al. first proposed a family of authentication protocols (LMAP [10], EMAP [11], M^{2}AP [12]) that only involve simple bit-wise operations, such as XOR, AND, OR, and addition modulo 2^m ($\bmod 2^m(+)$) for ultra-lightweight RFID authentication protocols in 2006. However, these protocols are vulnerable to both active and passive attacks [13,14]. Chien et al. proposed the SASI protocol [15], an ultra-lightweight protocol which defines a new bitwise transformation operation, the left cyclic shift operation in 2007. However, the weaknesses of SASI protocols were analyzed in [16,17], respectively, and the results showed that it suffers from the tracking attack, the key leakage attack, and the denial-of-service attack. Tian et al. proposed a RAPP protocol based on $Per(X, Y)$ operation in [18] in 2012; however, the protocol is vulnerable to the desynchronization attack [19], as well as cannot be resistant to the impersonation attacks and full exposure of its secret information by eavesdropping on the collected communication messages [20].

In recent years, numerous research efforts on Physically Unclonable Functions (PUF)-based RFID secure authentication protocols have been proposed. Pappu et al. [21] first introduced the concept of PUFs in 2002, which are based on physical systems to achieve one-way outputs. PUFs as an emerging lightweight hardware security primitive which can provide a secure lightweight cryptographic computational for resource-constrained devices such as RFID has received increasing concern. The idea of the PUF-based authentication protocol is to employ the PUF module to pre-generate a large number of Challenge-Response Pairs (CRPs), and the CRPs are stored in the server's database. The existing PUF-based RFID secure authentication protocols are based on pre-computing multiple sets of CRPs and storing them in a reader (server) database, which makes them vulnerable to the machine learning attacks [25].

Akgün et al. [23] proposed a PUF-based authentication scheme for RFID systems, however, the scheme could not ensure the forward security [9] in 2015. Liang et al. [24] proposed a double PUF-based RFID identity authentication protocol in service-centric internet of things environments, however, the scheme suffers from the possibility of tag counterfeiting and cannot robust against the desynchronization attacks (Denial-of-Service attacks, DoS) [25] and the replay attacks [22] in 2019. Gope et al. [6] proposed a PUF-based RFID secure authentication protocol for UAV applications in 2021. Multiple sets of CRPs are pre-generated and stored in the server during the setup phase to accomplish the subsequent process PUF-based RFID authentication communication. In addition, the storage cost of tags is too expensive and vulnerable to the potential multi-round desynchronization attacks in this protocol (See Appendix A for detailed analysis).

Our Contributions. In this paper, we focus on the requirements of security properties and the potential malicious attacks in RFID systems, as well as the vulnerabilities from previous research. To solve the vulnerability of RFID systems to various malicious attacks we propose a secure lightweight RFID mutual authentication protocol based on the configurable tristate physical unclonable

functions (CT PUF) and the cryptographic primitive of verifiable secret sharing (VSS). The main contributions of this paper are summarized below:

- This paper flexibly combines three techniques, CT PUF, VSS and Hash Table, on which a secure lightweight authentication protocol is proposed. The proposed protocol addresses the performance and security issues of RFID authentication protocols, while low-cost RFID tags can achieve a compromise between security and performance. More specifically, the server takes advantage of VSS to enable that it does not require the storage of explicit challenge-response pairs for CT PUF but instead stores a secret share of $CRPs_{Arbiter_T}$. The tags are equipped with CT PUF structures to ensure more security against the machine learning attacks.
- In previous studies, it is necessary for the server to perform an exhaustive search when identifying the tag. In particular, there is no fine scalability with a large number of the tags. For each record entry of the tag's secret parameters, our protocol is not to store them directly in chronological order but to take advantage of the hash table storage mechanism. This mechanism can effectively reduce the complexity of search time. The time complexity of the server to perform the search under ideal conditions is $\mathcal{O}(1)$ only. Therefore, it can effectively reduce the time overhead of the server in searching and updating the tag's secret parameter.
- The formal analysis of the proposed protocol using BAN logic shows that the correctness of authentication communication between the server and the tag can be guaranteed. We also verify the security of the proposed protocol using the Scyther, and the simulation results show that no attacks are detected. In addition, We analyze security properties and resistance to malicious attacks. The results show that our protocol satisfies various security properties such as data integrity, data confidentiality, mutual authentication, anonymity, forward security and is resistant to malicious attacks such as the man-in-the-middle attacks, the replay attacks, the desynchronization attacks, and the physical attacks and cloning attacks, the impersonation attacks.

1.1 Structure of the Paper

The rest of this paper is organized as follows. The underlying preliminaries are introduced in Section 2. The proposed CT PUF and VSS-based lightweight RFID authentication protocol is presented detailed in Sect. 3. The informal security and formal security of the proposed scheme is analyzed in Sect. 4 and Sect. 5, respectively. Then, in Sect. 6, we give the performance evaluation of the proposed scheme. Finally, we present the conclusion of the paper and look to the future in Sect. 7.

2 Preliminaries

In this section, we present a brief description of the background required to build the proposed protocol. Due to space constraints, the detailed description

of physical unclonable functions and Feldman's (t, n)-threshold verifiable secret sharing scheme are presented in Appendix B and C, respectively.

Here, we summarize the security properties and potential malicious attacks by RFID system security requirements as well as other researches [6,9,23,24], as follows.

- **Data Integrity.** It refers to the integrity of the communication messages during the authentication process as well as the secret parameters stored by the reader and tag. The proposed authentication scheme should provide an integrity confirmation mechanism that can verify whether the communication message has been tampered with by an adversary.
- **Data Confidentiality.** It refers to the communication message transmitted between the reader (the server), and the tag should be a ciphertext message generated by obfuscating or encrypting secret parameters. Hence, even if the adversary intercepts the message by means of eavesdropping, etc., it is impossible to obtain valuable information from it.
- **Anonymity.** It is mainly for tags, which have a unique identifier. Once that identity is captured by an adversary, then the tag can be continuously identified and tracked with that identity.
- **Mutual authentication.** Since there is a forgery attack in the potential attack, where the adversary impersonates as a reader or disguises as a tag. Therefore, the reader and tag are not trusted by each other in the authentication process, and mutual authentication between them is a necessary requirement for RFID authentication protocols.
- **Forward security.** Satisfying forward security means that even though all authentication messages between the reader and the tag during the authentication process are intercepted by the adversary, no authentication information or other secret parameters of the previous authentication session can be inferred from these messages.
- **Man-in-Middle attack.** This attack is undetectable to the readers and the tags; thus, the adversary can take control of the communication channel without being detected. The adversary establishes the communication channel with the reader and the tag respectively and intercepts the communication channel between the reader and the tag simultaneously. In this way, the communication messages in the system are forwarded by the adversary.
- **Replay attack.** This attack is an active attack that an adversary intercepts and replays communication messages during the authentication session. The adversary first eavesdrops on the communication channel, then he/she waits for authentication communication between a valid reader and a tag, and finally intercepts and holds the authentication session messages. In the future, the adversary impersonates as a legal reader or tag and replays the authentication messages for authentication.
- **Desynchronization attack.** The purpose of a desynchronization attack is to desynchronize shared key parameters that the reader and the tag require synchronization, and the attack occurs during the final update phase of a legitimate reader and a tag authentication communications.

- **Impersonation attack.** The adversary impersonates a legitimate tag or server by counterfeiting the tag's or server's identity and secret parameters, or even utilizing the forged parameters to generate authentication messages for authenticated communication in order to spoof the valid tag or server as a legitimate one.
- **Cloning attack.** This attack is a malicious attack on a tag in which an adversary clones a completely identical tag by detecting the tag's circuit or trying other mechanical methods to analyze the tag's circuit. Further, once the attack is successful, the adversary can initiate other malicious attacks.

3 The Proposed Protocol

In this section, we first explain the relevant notations and descriptions involved in our protocol. Next, the system model and the adversary model are described. Then the assumptions that the proposed scheme relies on are depicted. Finally, our protocol composes of the initialization phase and the authentication phase, which are described in detail step by step according to the authentication sequence.

3.1 Notations

The notations and descriptions involved in the protocol are detailed listed in Table 1.

3.2 Assumptions of Our Proposed Protocol

The basic assumptions followed in our protocol are as follows.

(a) The reader communicates with the back-end server over a secure channel, and therefore consider both as one unit. The reader (server) communicates with the tag over an open and insecure wireless channel, which is vulnerable to various malicious attacks, such as the man-in-middle attacks, the impersonation attacks and the replay attacks.
(b) The tag is a low-cost device with limited computing and storage resources, i.e., a low-cost passive RFID tag. Therefore, the tag can only perform lightweight or ultra-lightweight operations.
(c) The PUFs module is embedded in the Tag, and the PUFs structure refers specifically to the new Configurable Tristate PUF (CT PUF) proposed by Zhang et al. in [29]. The CT PUF can effective against a variety of machine learning attacks, such as logistic regression (LR), support vector machines (SVM), covariance matrix adaptation evolutionary strategies (CMA-ES), and artificial neural networks (ANN).

Table 1. Notations and descriptions

Notation	Descriptions	Notations	Descriptions
T	RFID Tags	S	Server and reader unit
CT $\text{PUF}_T(\cdot)$	CT PUF module embedded in T	$M_{\text{CT PUF}_T}$	A soft model of CT PUF_T trained on S
SID_T^{i-1}	$i-1$ round shadow identity of T	SID_T^i	i round shadow identity of T
SID_T^{i+1}	$i+1$ round shadow identity of T	$PRNG(\cdot)$	Pseudo-random number generator
$C_{Arbiter_T}$	Stable challenge of CT $\text{PUF}_T(\cdot)$	$R_{Arbiter_T}$	The stable response of CT $\text{PUF}_T(\cdot)$
C_{XOR_i}	The challenge of Tristate PUF in the bitwise XOR obfuscation mechanism	R_{XOR_i}	Temporary response of Tristate PUF in the bitwise XOR obfuscation mechanism
C_i	i round random challenge of CT $\text{PUF}_T(\cdot)$	R	The real response of CT $\text{PUF}_T(\cdot)$
$\hat{R}$	An obfuscated response of $M_{\text{CT PUF}}$	τ	Difference threshold of $\text{FHD}(R,\hat{R})$
N_S	The random number generated by S	N_T	The random number generated by T
$M_i, i = \{1,2,3,4\}$	Authentication message	S_1, S_2	Shares of the $R_{Arbiter_T}$
K_T^{i-1}	$i-1$ round secret key of T	K_T^i	i round secret key of T
K_T^{i+1}	$i+1$ round secret key of T	K_S	The secret key of S
$Index_T$	T's storage index in database	$h(\cdot)$	A lightweight cryptographic Hash functions
$\|$	String conjunction operations	$\bigoplus$	Bitwise XOR
$\mathsf{VSS.Share}(\cdot)$	A distribution algorithm for verifiable secret sharing	$\mathsf{ht}(\cdot)$	The Hash algorithm that generates $Index_T$
$\mathsf{VSS.Rec}(\cdot)$	A reconstruction algorithm for verifiable secret sharing	$\mathsf{VSS.Verify}(\cdot)$	A verification algorithm for verifiable secret sharing
$\alpha = \{\alpha_i\}, i = \{1,\cdots,t-1\}$	The commitments of polynomial coefficients	$\mathsf{FHD}(\cdot)$	The Fuzzy Hamming Distance
$\mathsf{Sea}(\cdot)$	A function to find string information	$\mathsf{match}(\cdot)$	A function to match strings
$\mathsf{Pad}(\cdot)$	A function for bit-wise padding	$P(\cdot)$	PUF module embedded in T

3.3 Authentication Process

In this subsection, we fist present the authentication process of the proposed protocol based on CT PUF and the cryptographic primitive of verifiable secret sharing. The whole authentication process is divided into two phases: the setup phase and the authentication phase.

Setup Phase. During the setup phase server (reader) unit and tag communicate with each other over a secure channel in a secure environment. This phase completes the initialization and sharing of secret parameters between server and tag. For a tag's CT PUF$_T$, it has some stable $CRPs : CR_{Arbiter} = \{C_{Arbiter}, R_{Arbiter}\}$, assuming that server has acquired and stored the $CRPs$ during the initialization phase. The server register a tag's CT PUF$_T$ by collecting R_{XOR} and training a soft model $M_{\text{CT PUF}_T}$ for each tag's CT PUF$_T$. The detailed process of communication interaction during the setup phase is depicted in Fig. 2. The detailed initialization steps of the proposed protocol are as follows.

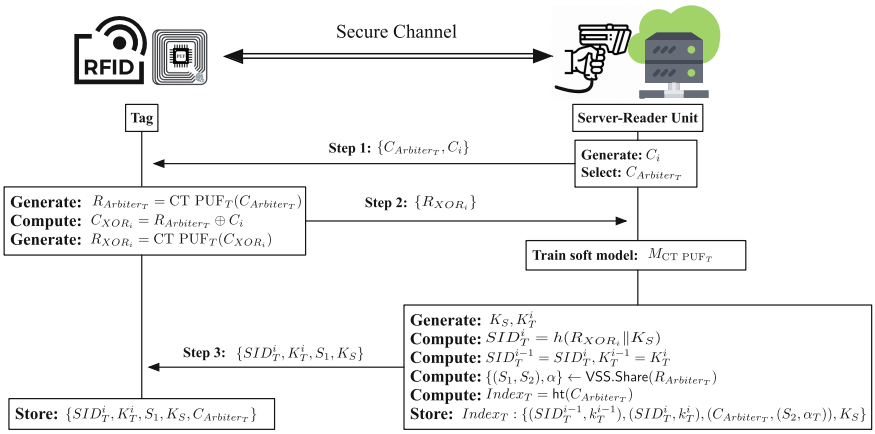

Fig. 2. Setup phase of the proposed protocol.

Step1: The server sends messages to the tag as $S \xrightarrow{\{C_{Arbiter_T}, C_i\}} T$

(1) First, S randomly selects a pair of $CR_{Arbiter_T}$ from some stable $CRPs$ of the CT PUF$_T$, and generates a random challenge C_i for the current round session.
(2) Then S sends $\{C_{Arbiter_T}, C_i\}$ to T.

Step2: The tag sends messages to the server as $T \xrightarrow{\{R_{XOR_i}\}} S$

(1) By default, the CT PUF_T of a unenroled T is in the Arbiter PUF working state. Once T receives the message $\{C_{Arbiter_T}, C_i\}$, it first takes $C_{Arbiter_T}$ as the input to the CT PUF_T, and then generates the response $R_{Arbiter_T} =$ CT $\text{PUF}_T(C_{Arbiter_T})$.
(2) Then, the real input challenge $C_{XOR_i} = R_{Arbiter_T} \oplus C_i$ of CT PUF_T is computed by the obtained $R_{Arbiter_T}$ to XORing C. Then, the C_{XOR_i} as the input of CT PUF to obtain the $R_{XOR_i} =$ CT $\text{PUF}_T(C_{XOR_i})$ in Arbiter PUF, RO PUF or BR PUF working states, T sends R_{XOR_i} to S.
(3) Finally, T sends $\{R_{XOR_i}\}$ to S.

Step3: The server sends messages to the tag as $S \xrightarrow{\{SID_T^i, K_T^i, (C_{Arbiter_T}, S_1), K_S\}} T$

(1) Once S receives R_{XOR_i} and then S uses R_{XOR_i} to train a software model $M_{\text{CT PUF}_T}$ for CT PUF_T.
(2) Next, S generates a secret i-round session key K_T^i for T and a secret key K_S for itself. Then, S computes the shadow identity $SID_T^i = h(R_{XOR_i} \| K_S)$.
(3) To keep off the desynchronization attacks, S stores secret parameters shared with T for two authentication session. Therefore, S performs the following computing: $SID_T^{i-1} = SID_T^i, K_T^{i-1} = K_T^i$.
(4) To prevent the $R_{Arbiter_T}$ of CT PUF from being directly accessed after setup pahse, we make advantage of the verifiable secret sharing scheme with $(2, 2)$ threshold to store $R_{Arbiter_T}$ instead of the explicit storage. S computes: $\{(S_1, S_2), \alpha_T\} \leftarrow \mathsf{VSS.Share}(R_{Arbiter_T})$:
   ① S generates two primes numbers p and q such that $q|(p-1)$, and $g \in \mathbb{Z}_p*$ an element of order q.
   ② S generates the random polynomial $P(x) = a_0 + a_1 x$ over $\mathbb{Z}_q$ such that $a_0 = R_{Arbiter_T}$, and makes commitment $\alpha_T = \{\alpha_i = g^{a_i} \bmod p, i = 0, 1\}$.
   ③ S computes $S_1 = P(1)$ and $S_2 = P(2)$, where S_1 is sent securely to T after with the other parameters,S_2 is stored by itself.
(5) When all secret parameters are generated, S takes advantages of hash table to store these parameters as a record entry. Here, the storage index $Index_T = \mathsf{ht}(C_{Arbiter_T})$ is generated by the $C_{Arbiter_T}$. The storage structure of S is shown in Fig. 3.
(6) Then, S stores: $Index_T$:$\{(SID_T^{i-1}, k_T^{i-1}), (SID_T^i, k_T^i), (C_{Arbiter_T}, (S_2, \alpha_T)), K_S\}$ and sends $\{SID_T^i, K_T^i, S_1, K_S\}$ to T.
(7) Once T receives $\{SID_T^i, K_T^i, S_1, K_S\}$, then T stores $\{SID_T^i, K_T^i, S_1, K_S, C_{Arbiter_T}\}$

Authentication Phase. After the initialization phase, the tag enters an open insecure real-world usage scenario in which S and T communicate over an insecure wireless channel. Therefore, the security and privacy of authentication during communication need to be guaranteed as the communication interaction

$Index_{T_k} = \mathsf{ht}(C_{Arbiter_{T_k}})$	$\{(SID_{T_k}^{i-1}, K_{T_k}^{i-1}), (SID_{T_k}^{i}, K_{T_k}^{i}), (C_{Arbiter_{T_k}}, S_{T_{k2}}), K_S\}$
Null	Null
⋮	⋮
Null	Null
$Index_{T_i} = \mathsf{ht}(C_{Arbiter_{T_i}})$	$\{(SID_{T_i}^{i-1}, K_{T_i}^{i-1}), (SID_{T_i}^{i}, K_{T_i}^{i}), (C_{Arbiter_{T_i}}, S_{T_{i2}}), K_S\}$
⋮	⋮
$Index_{T_j} = \mathsf{ht}(C_{Arbiter_{T_j}})$	$\{(SID_{T_j}^{i-1}, K_{T_j}^{i-1}), (SID_{T_j}^{i}, K_{T_j}^{i}), (C_{Arbiter_{T_j}}, S_{T_{j2}}), K_S\}$

Fig. 3. The storage structure of S.

between S and T is vulnerable to various malicious attacks, such as Man-in-Middle attack, desynchronization attack, replay attack, etc. The detailed process of communication interaction during the authentication phase is depicted in Fig. 4. The detailed authentication steps of the proposed protocol are as follows.

Step1: The tag sends messages to the server as $T \xrightarrow{M_1:\{SID_T^i, C_T, N_T^*\}} S$

(1) Each new round of authentication session is initiated by T. T first selects SID_T and K_S, and generated random nonce N_T by $PRNG(\cdot)$ for the i-round authentication session, respectively.
(2) Next, T computes $N_T^* = N_T \oplus K_S$, $C_T = C_{Arbiter_T} \oplus N_T$.
(3) Finally, T sends a messages $M_1 : \{SID_T^i, C_T, N_T^*\}$ to S.

Step2: The server sends messages to the tag as $S \xrightarrow{M_2:\{C_i, N_S^*, Res_S\}} T$

(1) Once S receives message $M_1 : \{SID_T^i, C_T, N_T^*\}$, it first extracts the random number $N_T = N_T^* \oplus K_S$ generated by T from N_T^* and then uses N_T to extract $C_{Arbiter_T} = C_T \oplus N_T$ from C_T.
(2) Then, S computes the stored index $Index_T = ht(C_{Arbiter_T})$ of T using $C_{Arbiter_T}$, then reads the record entry$\{(SID_T^i, K_T^i), S_2\}$. Note that if the SID_T^i in the read record entry does not match the one received, then continue reading from the index until a match is found.
(3) Next, S randomly generates a challenge C_i and random number N_S by $PRNG(\cdot)$, and then computes $N_S^* = K_T^i \oplus N_S$, $Res_S = h(C_i, N_T \| K_T^i \| N_S^*)$.
(4) Finally, S sends a messages $M_2 : \{C_i, N_S^*, Res_S\}$ to T.

Step3: The tag sends messages to the server as $T \xrightarrow{M_3:\{R_T, S_{T_x}, S_{T_y}, Res_T\}} S$

(1) Once T receives $M_2 : \{C_i, N_S^*, Res_S\}$, it first checks the integrity of M_2. T computes $Res'_S = h(C_i \| N_T \| K_T^i \| N_S^*)$ using the received N_S^* and C_i and its locally stored K_T^i, N_T. The integrity of Res_S is verified by checking the equation $Res'_S \stackrel{?}{=} Res_S$. If the equation holds, the expected S is authenticated, and then T continues with the subsequent authentication process. Otherwise, Res_S can not be authencated by T and the protocol terminates.

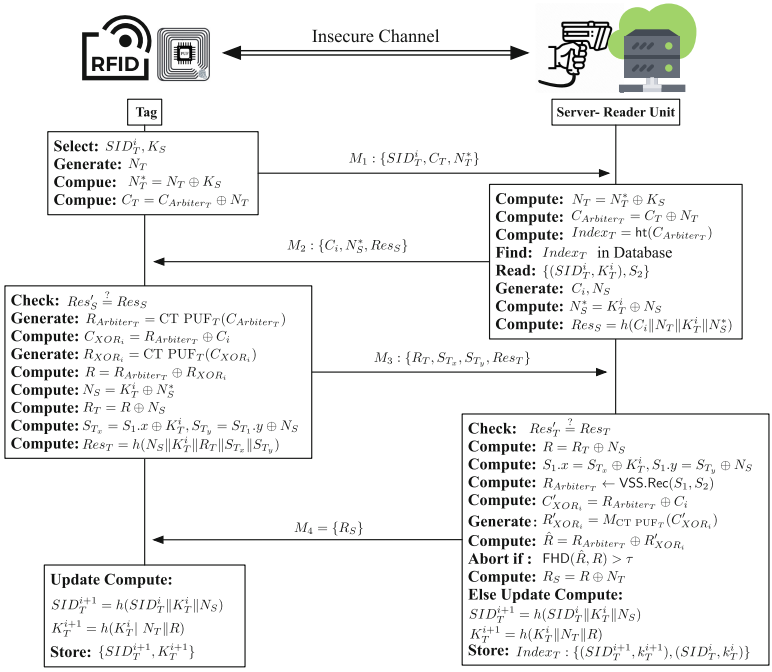

Fig. 4. Authentication phase of the proposed protocol.

(2) Then, T first takes $C_{Arbiter}$ as the input of the CT PUF_T to generate the response $R_{Arbiter_T} = \text{CT PUF}_T(C_{Arbiter_T})$. Then the tag's CT PUF_T Computes the real input challenge $C_{XOR_i} = R_{Arbiter_T} \oplus C_i$. And then, the C_{XOR_i} as the input to obtain the $R_{XOR_i} = \text{CT PUF}_T(C_{XOR_i})$. Then, the real response $R = R_{Arbiter_T} \oplus R_{XOR_i}$ of CT PUF_T is generated.
(3) Next, T extracts the random number $N_S = K_T^i \oplus N_S^*$ generated by S from N_S^* and then computes $R_T = R \oplus N_S$, $S_{T_x} = S_1.x \oplus K_T^i$, $S_{T_y} = S_{T_1}.y \oplus N_S$ and $Res_T = h(N_S \| K_T^i \| R_T \| S_{T_x} \| S_{T_y})$.
(4) Finally, T sends a messages $M_3 : \{R_T, S_{T_x}, S_{T_y}, Res_T\}$ to S.

Step4: The server sends messages to the tag as $S \xrightarrow{M_4:\{R_S\}} T$

(1) Once S receives $M_3 : \{R_T, S_{T_x}, S_{T_y}, Res_T\}$, it first checks the integrity of M_3. S computes $Res_T' = h(N_S \| K_T^i \| R_T \| S_{T_x} \| S_{T_y})$ using the received R_T,S_{T_x},S_{T_y} and its locally stored K_T^i, N_S. The integrity of Res_T is verified by checking the equation $Res_T' \stackrel{?}{=} Res_T$. If the equation holds, the expected T is authenticated, and then S continues with the subsequent authentication process. Otherwise, Res_T can not be authencated by S and the protocol terminates.
(2) Then, S extracts the real response $R = R_T \oplus N_S$ of CT PUF_T from R_T, and then, S computes $R_T = R \oplus N_S$, $S_1.x = S_{T_x} \oplus K_T^i, S_1.y = S_{T_y} \oplus N_S$.

(3) Next, S reconstruct the $R_{Arbiter}$ by the obtained $S_1 = (S_1.x), S_1.y)$ and S_2 stored by itself.
   (a) Note that the validity of S_1 needs to be verified by $\mathsf{VSS.Verify}(\cdot)$. S can verify the correctness of the share S_1 by checking the equation $g^{S_1} \bmod p \stackrel{?}{=} \prod_{j=0}^{1} \alpha_j^{1^j} \bmod p$.
   (b) If the equation holds, then S_1 is valid and S could reconstruct $R_{Arbiter}$. Otherwise it is fail, and the protocol terminates.

(4) Next, S computes $C'_{XOR_i} = R_{Arbiter_T} \oplus C_i$, and then takes C'_{XOR_i} as input of the trained soft model $M_{\text{CT PUF}_T}$ to generates $R'_{XOR_i} = M_{\text{CT PUF}_T}(C'_{XOR_i})$, then computes $\hat{R} = R_{Arbiter_T} \oplus R'_{XOR_i}$. And then, S computes the hamming distance $\mathsf{FHD}(R, \hat{R})$. If $\mathsf{FHD}(R, \hat{R}) \leq \tau$ holds, T is verified successfully by S, otherwise it is fail and the protocol terminates.

(5) Then, S computes $SID_T^{i+1} = h(SID_T^i \| K_T^i \| N_S)$, $K_T^{i+1} = h(K_T^i \| N_T \| R)$ for the next authentication session round, and then stores $Index_T$: $\{(SID_T^{i+1}, K_T^{i+1}), (SID_T^i, K_T^i)\}$.

(6) Then, S computes $R_S = R \oplus N_T$ and sends a message $M_4 : \{R_S = R \oplus N_T\}$ to T.

(7) Once T receives $M_4 : \{R_S = R \oplus N_T\}$, it first extracts $R' = R_S \oplus N_T$ from R_S, and checks the correctness of $R' \stackrel{?}{=} R$. If it holds, then T computes $SID_T^{i+1} = h(SID_T^i \| K_T^i \| N_S)$, $K_T^{i+1} = h(K_T^i | \ N_T \| R)$ for the next round authentication session, and stores $\{SID_T^{i+1}, K_T^{i+1}\}$. Otherwise, T terminates the protocol and does not update the key parameters.

4 BAN Logic Analysis and Simulation Using the Scyther

In this section, we analyze the correctness and security of the proposed protocol using the BAN [31] logic formally, and simulate the proposed protocol using the Scyther informally. Note that we only analyze the correctness and security of communication for the authentication phase and not for the initialization phase which is in a secure environment.

4.1 BAN Logical Analysis

Authentication protocol is described by enumerating their communication messages, so the protocol communication sessions are first described formally in the BAN logic language. Inevitably, the protocol must make initial assumptions and build the set of initial assumptions. Then, the communication messages of the protocol are converted into formal formulas that the BAN logic can recognize to create an idealized protocol model. Next, the proving goals to be proved are specified. Finally, these proving goals are proved according to the BAN logical postulates. The BAN logical formal analysis of the proposed protocol is specified as follows.

Basic Notations and Logical Postulates. The basic BAN-logic proof notations as shown in Appendix D.1 Table 7. The five BAN logical postulates are used when we analysis the protocol, as shown in Appendix D.2.

Idealized Model of the Protocol. According to the basic notation of BAN logic listed in Table 7, the authentication communication interaction messages of the proposed protocol are converted into an idealized model. There are four authentication message: **M1**: $S \triangleleft \{SID_T^i, \{C_T\}_{N_T}, \{N_T^*\}_{K_S}\}$, **M2**: $T \triangleleft \{C_i, \{N_S^*\}_{K_T^i}, \{Res_S\}_{K_T^i}\}$, **M3**: $S \triangleleft \{\{R_T\}_{N_S}, \{S_T\}_{K_T^i}, \{Res_T\}_{K_T^i}\}$. **M4**: $T \triangleleft \{R_S\}_{N_T}\}$, With the BAN logical postulates R4, we have the idealized model of session messages between S and T in the protocol. The detailed session message are listed in the Table 2.

Table 2. The idealized model of the protocol

Message	Message	Message
M1: $S \triangleleft \{N_T^*\}_{K_S}$	**M2**: $S \triangleleft \{C_T\}_{N_T}$	**M3**: $S \triangleleft \{Res_S\}_{N_T, K_T^i}$
M4: $T \triangleleft \{N_S^*\}_{K_T^i}$	**M5**: $T \triangleleft \langle C_i \rangle_{K_T^i, N_T}$	**M6**: $S \triangleleft \{Res_T\}_{K_T^i, N_S}$
M7: $T \triangleleft \{R_T\}_{N_S}$	**M8**: $S \triangleleft \{S_{T_x}\}_{K_T^i}$	**M9**: $S \triangleleft \{S_{T_y}\}_{N_S}$
M10: $T \triangleleft \{R\}_{N_T}$		

Initiated Assumptions of Protocol. The basic initial assumptions followed in the protocol are shown in Table 3.

Table 3. The initial assumptions of the protocol

Assumption	Assumption	Assumption
A1: $S \mid\equiv S \overset{K_S}{\longleftrightarrow} T$	**A2**: $T \mid\equiv T \overset{K_S}{\longleftrightarrow} S$	**A3**: $S \mid\equiv S \overset{K_T^i}{\longleftrightarrow} T$
A4: $T \mid\equiv T \overset{K_T^i}{\longleftrightarrow} S$	**A5**: $S \mid\equiv \#(N_T)$	**A6**: $T \mid\equiv \#(N_S)$
A7: $S \mid\equiv \#(N_S)$	**A8**: $T \mid\equiv \#(N_T)$	**A9**: $S \mid\equiv T \mid\Rightarrow N_T^*$
A10: $S \mid\equiv T \mid\Rightarrow C_T$	**A11**: $T \mid\equiv S \mid\Rightarrow C_i$	**A12**: $T \mid\equiv S \mid\Rightarrow N_S^*$
A13: $T \mid\equiv S \mid\Rightarrow Res_S$	**A14**: $S \mid\equiv T \mid\Rightarrow R_T$	**A15**: $S \mid\equiv T \mid\Rightarrow S_{T_x}$
A16: $S \mid\equiv T \mid\Rightarrow S_{T_y}$	**A17**: $S \mid\equiv T \mid\Rightarrow Res_T$	**A18**: $S \mid\equiv T \mid\Rightarrow S_{T_y}$
A19: $T \mid\equiv S \mid\Rightarrow R_S$		

Proving Goals of the Protocol. The proving goals of the protocol are listed in Table 4. The detailed process of proving the goals is as follows.

– **For G1:** $S \mid\equiv N_T^*$**:** With the A1, M1, and R1, we have Result1: $S \mid\equiv T \mid\sim N_T^*$. Then, with the A5 and R5, we have Result2: $S \mid\equiv \#(N_T^*)$. Then, with the Result1, Result2 and R3, we have Result3: $S \mid\equiv T \mid\equiv N_T^*$. Finally, with A9, Result3 and R2, we have G1: $S \mid\equiv N_T^*$.

Table 4. The proving goals of the protocol

Proving goal	Proving goal	Proving goal	Proving goal
G1: $S \mid\equiv N_T^*$	**G2**: $S \mid\equiv C_T$	**G3**: $T \mid\equiv Res_S$	**G4**: $T \mid\equiv C_i$
G5: $T \mid\equiv N_S^*$	**G6**: $S \mid\equiv Res_T$	**G7**: $S \mid\equiv R_T$	**G8**: $S \mid\equiv S_{T_x}$
G9: $S \mid\equiv S_{T_y}$	**G10**: $T \mid\equiv R_S$		

- **For G2:** $S \mid\equiv C_T$**:** With the **G1**,M2 and R1, we have Result1: $S \mid\equiv T \mid\sim C_T$. Then, with the A5 and R5, we have Result2: $S \mid\equiv \#(C_T)$. Then, with the Result1, Result2 and R3, we have Result3: $S \mid\equiv T \mid\equiv C_T$. Finally, with A10, Result3 and R2, we have G2: $S \mid\equiv C_T$.
- **For G3:** $T \mid\equiv Res_S$**:** With the A4, M3 and R1, we have Result1: $T \mid\equiv S \mid\sim Res_S$. Then, with the A8, and R5, we have Result2: $T \mid\equiv \#(Res_S)$. Then, with the Result1, Result2 and R3, we have Result3: $T \mid\equiv S \mid\equiv Res_S$. Finally, with A13, Result3 and R2, we have G3: $S \mid\equiv Res_S$.
- **For G4:** $T \mid\equiv C_i$,**G5:**$T \mid\equiv N_S^*$**:** The security of G4 and G5 is equivalent to that of G5, so as long as the security of G3 is proved, the security of G4 and G5 is also proved.
- **For G6:** $S \mid\equiv Res_T$**:** With the A3, M6 and R1, we have Result1: $S \mid\equiv T \mid\sim Res_T$. Then, with the A7, and R5, we have Result2: $T \mid\equiv \#(Res_T)$. Then, with the Result1, Result2 and R3, we have Result3: $S \mid\equiv T \mid\equiv Res_T$. Finally, with A17, Result3 and R2, we have G6: $S \mid\equiv Res_T$.
- **For G7:** $S \mid\equiv R_T$, **G8:** $S \mid\equiv S_{T_x}$, **G9:** $S \mid\equiv S_{T_y}$**:** The security of G7, G8 and G9 is equivalent to that of G6, so as long as the security of G6 is proved, the security of G7, G8 and G9 is also proved.
- **For G10:** $T \mid\equiv R_S$**:** With the AX, M10 and R1, we have Result1: $T \mid\equiv S \mid\sim R_S$. Then, with the A8, and R5, we have Result2: $T \mid\equiv \#(R_S)$. Then, with the Result1, Result2 and R3, we have Result3: $T \mid\equiv S \mid\equiv R_S$. Finally, with A19, Result3 and R2, we have G10: $S \mid\equiv R_S$.

4.2 Simulation Using the Scyther

The Security Protocol Description Language (SPDL) description for function and messages, the SPDL description for role S and for role T in Appendix E. The verification result of Scyther simulation as shown in Appendix E Fig. 6.

5 Security Analysis

In Sect. 2, we provide the security requirements and potential attacks of the RFID system. In this section, we present an detailed analysis of the security of the proposed protocol according to the above security requirements.

Proposition 1 (Data Integrity). *The proposed protocol provides an effective verification mechanism for the data integrity of communication interaction messages.*

Proof. The data integrity of the communication message is an evidence to verify that the session information has not been tampered with by the adversary. The mutual authentication communication messages $M_2 : \{C_i, N_S^*, Res_S\}$, $M_3 : \{R_T, S_{T_x}, S_{T_y}, Res_T\}$ between S and T not only hide the key parameters, but also the Res_S and Res_T provide assurance for the integrity of M_1 and M_2, respectively. $Res_S = h(C_i \| N_T \| K_T^i \| N_S^*)$ is a hash value which is obtained by hashing $C_i \| N_T \| K_T^i \| N_S^*$, where C_i,N_T^* are part of M_1, and K_T^i and N_T are the parameters shared by S and T. Relying on the one-way nature of the hash function, the adversary cannot efficiently compute the matching Res_S when it can only tamper with C_i and N_S^* but does not gain access to K_T^i and N_T. For $Res_T = h(N_S \| K_T^i \| R_T \| S_{T_x} \| S_{T_y})$, the same applies. Thus, the integrity of the communication messages during mutual authentication of S and T is efficiently detected whenever they suffer from tampering damage. Therefore, we claim that the proposed protocol can effectively ensure data integrity.

Proposition 2 (Data Confidentiality). *The proposed protocol is able to ensure the confidentiality of the communication messages.*

Proof. Our protocol contains four communication messages, $M_1 : \{SID_T^i, C_T, N_T^*\}$, $M_2 : \{C_i, N_S^*, Res_S\}$, $M_3 : \{R_T, S_{T_x}, S_{T_y}, Res_T\}$ and $M_4 : \{R_S\}$. All messages except SID_T^i and C_i are transmitted after obfuscated encryption, See Sect. 3.3 for details. In addition, the obfuscated key parameters are shared only by the legitimate T and S, so the adversary cannot obtain valid secret data through the authentication message. Therefore, the proposed protocol can guarantee the data confidentiality of communication messages.

Proposition 3 (Anonymity). *The proposed protocol utilizes an updated pseudonymous identity mechanism to effectively protect the identity privacy of the tag, and an adversary cannot infer the true identity of the tag through authentication messages.*

Proof. Anonymity also means untraceability, and T does not specify a unique identity in our protocol. The shadow identity SID_T^i is only a momentary value in communication session round i. $SID_T^{i+1} = h(SID_T^i \| K_T^i \| N_S)$ will be updated according to the newly generated random number N_S and key parameters K_T^i as long as the authentication communication is successfully executed. The adversary cannot infer the tag's identity from the session messages M_1, M_2, M_3, M_4. In particular, according to the update mechanism SID_T^i of M_1 in two different rounds (SID_T^i,$SID_T^{i+1}|$) are distinct and without direct correlation. Once T receives $M_4 = \{R_S\}$ and the contained R is successfully checked, then T computes $SID_T^{i+1} = h(SID_T^i \| K_T^i \| N_S)$, $K_T^{i+1} = h(K_T^i | \ N_T \| R)$, and stores $\{SID_T^{i+1}, K_T^{i+1}\}$. Therefore, the adversary cannot identify T based on the SID_T^i and performs constant tracking attacks.

Proposition 4 (Mutual Authentication). *It is crucial in detecting adversaries disguised as legitimate entities in untrustworthy environments. The proposed protocol implements mutual authentication between S and T. The authentication terminates once one party cannot be authenticated by the other party.*

Proof. The **Step3 and Step4 of the authentication phase** in our protocol accomplish S and T mutual authentication. In Step3, T validates the validity of S by computing $Res'_S = h(C_i \| N_T \| K_T^i \| N_S^*)$, where N_S^* and C_i are received messages, K_T^i and N_T are locally stored shared key. As only a legitimate S holds the key parameter K_T^i and N_T shared with T to generate the matching message $Res_S = h(C_i \| N_T \| K_T^i \| N_S^*)$ with Res'. Therefore, if the equation $Res'_S \stackrel{?}{=} Res_S$ verifies successfully, it means that T verifies S as a legitimate server. Similarly, in step4 S first verifies whether the equation $Res'_T \stackrel{?}{=} Res_T$ holds or not. If it holds, then S has access to the secret share used to reconstruct $\hat{R}$ and the real response R of CT PUF_T. Then the final response $\hat{R}$ of $M_{\mathrm{CT\ PUF}_T}$ is generated. Finally, the validity of R is determined by comparing the hamming distance $\mathsf{FHD}(R, \hat{R})$. Obviously, according to the natural characteristics of CT PUF_T, only a legitimate tag can generate R that satisfies $\mathsf{FHD}(R, \hat{R}) \leq \tau$. If $\mathsf{FHD}(R, \hat{R}) \leq \tau$ verifies successfully, it means that S verifies T as a legitimate Tag. Therefore, the proposed protocol achieves mutual authentication of S and T.

Proposition 5 (Forward Security). *The interactive session messages for the authenticated communication of the proposed protocol ensure forward security.*

Proof. Our protocols have remarkable characteristics to ensure forward security. Authenticated communication of all messages are generated with a fresh random number N_S or N_T; moreover, S and T have not stored these nonces. Therefore, an adversary who intercepts the communication message but has no access to the random numbers N_S or N_T and key parameters K_T^i, K_S cannot effectively guess the exact valid key parameters.

Proposition 6 (Against the Man-in-Middle Attack). *The proposed protocol ensures security even if the authenticated communication is subject to the man-in-the-middle attacks.*

Proof. As we analyzed in the **Data Integrity** security property, our protocol provides verification of the integrity of communication messages. The integrity of the communication interaction messages M_2 and M_3 during mutual authentication between S and T can be effectively verified. For M_2, sent by the legitimate S suffers from a man-in-the-middle attack and is tampered with as $M'_2 : \{C'_i, N_S^{*\prime}, Res'_S\}$. In our protocol, M_2 contains three messages C_i,N_S^* and Res_S. The adversary can only effectively tamper with messages C_i,N_S^* and not Res_S, because the hash function ensures the security of Res_S. In turn, $Res_S = h(C_i, N_T \| K_T^i \| N_S^*)$ is generated by C_i,N_S^* as parameters, so C_i,N_S^*, and Res_S integrity are bound together. Hence, an adversary cannot tamper against the entire M_2 without being detected. Similarly, an adversary cannot tamper against the entire M_3 without being detected. Therefore, our protocol can effectively resist man-in-the-middle attacks.

Proposition 7 (Against the Replay Attack). *The proposed protocol ensures that a round of communication messages can only be validly applied for the authentication round. The adversary cannot replay the captured communication messages to spoof T or S and be authenticated.*

Proof. In our protocol, the generation of all the authentication communication messages and the updating of secret parameters utilize random numbers N_S,N_T freshly generated by S and T in each round of the session. The freshness of the authenticated message is guaranteed using a new random number generated each round session so that the adversary cannot be authenticated by S or T by replaying the captured message. Therefore, our protocol can effectively resist replay attacks.

Proposition 8 (Against the Desynchronization Attack). *The proposed protocol can ensure proper authenticated communication in subsequent session rounds even if it suffers from the desynchronization attacks.*

Proof. In our protocol, to resist potential desynchronization attacks, T does not initiate the update immediately after sending the message M_3. Instead, T needs to wait for $M_4 : \{R_S\}$ sent by S and performs the update only after the correctness of $R' = R_S \oplus N_T$ extracted from M_4 is verified($R' \stackrel{?}{=} R$). The adversary blocks M_4 to launch a desynchronization attack. S has completed the update operation when sending M_4, but T will not be updated because it does not correctly receive M_4. To solve this problem, S stores not only the updated secret parameters $SID_T^{i+1} = h(SID_T^i \| K_T^i \| N_S)$, $K_T^{i+1} = h(K_T^i \| N_T \| R)$ for communication session round $i+1$ but also the secret parameter (SID_T^i, k_T^i) of the successful authentication round i. After suffering a desynchronization attack, S initiates a new authentication session round $i+1$ with T. Since T does not update synchronously with S, the SID_T^{i+1}(actually SID_T^i) sent by T in $M1$ cannot match the SID_T^{i+1} in S. At this point, S tries to match it with SID_T^i and can successfully match. Then S continues the subsequent authentication process. Finally, if the authentication communication round $i+1$ is executed completely and successfully, S and T will synchronously update. Therefore, our protocol can effectively resist desynchronization attacks.

Proposition 9 (Against the Cloning Attack). *The proposed protocol ensures the security of the T in case of the physical attacks or the cloning attacks.*

Proof. In our protocol, the PUF structure equipped by the tag is the CT PUF structure, which uses the bitwise XOR mechanism to hide the relationship between the real challenge and the real response. ($R_{Arbiter_T}$ = CT $\text{PUF}_T(C_{Arbiter_T})$, $C_{XOR_i} = R_{Arbiter_T} \oplus C_i$, R_{XOR_i} = CT $\text{PUF}_T(C_{XOR_i})$, $R = R_{Arbiter_T} \oplus R_{XOR_i}$) In addition, we do not explicitly store the $CRPs$ of CT PUF_T in S, but take advantage of VSS to store the $R_{Arbiter_T}$ in a decentralized method. (see Sect. 3.3 Setup Phase for detailed.) Therefore, our protocol can effectively resist cloning attacks (machine learning attacks).

Proposition 10 (Against the Impersonation Attack). *The proposed protocol ensures security against the impersonation attacks.*

Proof. Counterfeiting attacks are divided into tag impersonation and server impersonation. For S impersonation, the adversary first blocks the message

$M_2 : \{C_i, N_S^*, Res_S\}$, then personates to be a legitimate S' to communicate with T. In our protocol, if the adversary tries to impersonate S' by forging the message M_2', the output is "failed" when T checks $Res_S \stackrel{?}{=} Res_S'$, since the secret parameters N_T, K_T, N_S shared by S and T are hidden in M_2. The adversary does not have access to these parameters and relying on the one-way nature of the hash function Res_S cannot be modified. Similarly, for T impersonation, the adversary first blocks the message $M_3 : \{R_T, S_{T_x}, S_{T_y}, Res_T\}$, then personates to be a legitimate S' to communicate with T. The counterfeit message M_3 cannot be verified. Therefore, our protocol can resistance to impersonation attacks.

A detailed comparison of the security properties and resistance to some malicious attacks among our protocol and existing PUF-based RFID authentication protocols [6,23,24] is listed in the Table 5.

Table 5. Security comparison

Security properties	Akgün et al. [23]	Liang et al. [24]	Gope et al. [6]	Our protocol
SP1	●	●	●	●
SP2	●	●	●	●
SP3	●	●	●	●
SP4	●	●	●	●
SP5	○	●	●	●
SP6	●	●	●	●
SP7	●	○	●	●
SP8	◐	○	◐	●
SP9	○	●	○	●
SP10	●	○	●	●
SP11	●	●	○	●
SP12	●	○	○	●

Notes: **SP1**: Data Integrity; **SP2**: Data Confidentiality; **SP3**: Anonymity; **SP4**: Mutual Authentication; **SP5**: Forward Security; **SP6**: Against Man-in-Middle Attacks; **SP7**: Against Replay Attacks; **SP8**: Against Desynchronization Attacks; **SP9**: Against Cloning Attacks; **SP10**: Against Impersonation Attacks; **SP11**: Without explicit $CRPs$ in S(reader/verifier); **SP12**: Scalability;
●: satisfied SP; ◐: partially satisfied SP; ○: not satisfied SP;

6 Performance Evaluation

In this section, we compare our protocol with existing typical RFID authentication schemes such as Akgün et al. [23], Liang et al. [24], Gope et al. [6] in terms of cost of storage space for tags, communication overhead, and tag computation overhead for tags. The length of each parameter in our protocol is 96-bit.

The parameters stored by T in the proposed protocol include a shadow identity SID_T^i, a secret key K_T^i of T, a secret key K_S of S, a secret share S_1 of the secret $R_{Arbiter_T}$, and a challenge $C_{Arbiter}$ of CT PUF$_T$. As a result, the total storage cost of T is 580-bit.

In our protocol, the computation methods involved in the process of T generation of authentication messages and verification of communication messages contain four types, which are lightweight hash function operation $h(\cdot)$, pseudo-random number generator operation $PRNG(\cdot)$, XOR operation $\oplus$, and CT PUF response generation operation CT PUF$_T(\cdot)$.

To complete the mutual authentication between S and T, there are four interactive communication messages as $M_1 : \{SID_T^i, C_T, N_T^*\}$, $M_2 : \{C_i, N_S^*, Res_S\}$, $M_3 : \{R_T, S_{T_x}, S_{T_y}, Res_T\}$,and $M_4 : \{R_S\}$, which are transmitted during a complete round of authentication session. As a result, the total communication overhead to successfully complete a communication session between S and T is 1056-bit.

In summary, the detailed comparison results of the performance evaluation of the proposed protocol with other protocols [6,23,24] are listed in Table 6.

Table 6. Performance comparison

	Akgün et al. [23]	Liang et al. [24]	Gope et al. [6]	Our protocol
Cost of storage space for tags	512-bit	0-bit	*$(2L+2L\cdot n)$-bit	580-bit
Communication overhead	896-bit	2048-bit	*$8L$-bit	1056-bit
Computation overhead for tags	$4h(\cdot) + 2P(\cdot) + 1PRNG(\cdot)$	$2\oplus + 2P(\cdot) + 2PRNG + 1\mathsf{Pad}(\cdot) + 1\mathsf{Sea}(\cdot) + 1\mathsf{Match}(\cdot)$	$3 \oplus + 6h(\cdot) + 2P(\cdot) + 1PRNG(\cdot) + 1\mathsf{FE.Gen}(\cdot)$	$8 \oplus + 4h(\cdot) + 2\text{CT PUF}_T(\cdot) + 1PRNG(\cdot)$

Notes: *:Not specified, $L = m$-bit;

7 Conclusion

In this paper, we concentrate on RFID system security requirements and potential malicious attacks, and in particular, we analyze the performance and security of the scheme by Gope et al. We investigate and analyze that cannot provide all the security requirements of the described RFID system. To address these security vulnerabilities and performance flaws, we propose a secure lightweight authentication scheme based on CT PUF, VSS and hash table mechanism. In our protocol, the server takes advantage of the hash table mechanism to store the tag entries and thus avoid exhaustive search during query and update. And

with the help of VSS to achieve no explicit storage of $CRPs(CRPs_{Arbiter_T})$ at the server. The CT PUF utilizes the bit-wise XOR obfuscation mechanism to hide the mapping relationship between challenge and response pairs. Therefore, the CT PUF structure built into the tag can further enhance the security of the tag against machine learning attacks. In the future, we will further explore mutual authentication protocols based on distributed storage such as blockchain technology and further extend the application scenarios.

Appendix

A Analysis of the Gope et al.'s Scheme

In this section, we analyze the high storage cost of tag and the security vulnerabilities of the authentication scheme proposed by Gope et al. in [6],

A.1 Performance and Security Analysis of Gope's Scheme

In this subsection, we analyze the performance and security issues of the Gope et al.'s scheme [6]. Specifically, the excessive storage overhead of low-cost tags and potential asynchronous attacks are analyzed, respectively. A detailed process is provided below.

Performance Analysis. We observe that S generates a set of shadow identities and emergency key pairs (SID, K_{em}) for T in the initialization phase of the protocol proposed by Gope et al. Specifically, a set of key pairs is pre-stored in order to resist desynchronization attacks, which is actually n sets of shadow identity identities sid and secret keys sk. The key parameters stored in T include $\{(SID_T^i, sk)\}$ and $(SID, K_{em}) = \{(sid_1, k_{em1}), (sid_2, k_{em2}), \ldots, (sid_n, k_{em})\}$. Obviously, the pre-stored set of contingency key pairs directly increases the storage overhead of the tag. In order to make the scheme more robust against desynchronization attacks, a larger number of pre-stored emergency key pairs must be required. The storage overhead of tag increases linearly as the number of contingency key pairs increases, which is unfriendly to low-cost RFID tags. And we also note that S stores all the CRPs of tag's PUF module, which is also a potential threat for tag's PUF module to be subjected to machine learning attacks. We also note that S stores all the CRPs of the PUF module of tag, which is also a potential threat to the PUF module of tag, such as an adversary initiates a machine learning modeling attack by collecting CRPs. There have been research works on machine learning attacks against various PUF structures such as Arbiter PUFs, Configurable RO PUFs, SRAM PUFs, etc.

Security Analysis. The communication of the protocol suffers from a desynchronization attack, more specifically, the adversary blocks the session message M_3, and S cannot receive M_3 correctly. In this point, T performs the

update computation immediately after sending the session message M_3, which is blocked by the adversary so that S does not receive the message properly and does not perform any computation. The adversary successfully performs a desynchronization attack on S and T. The key parameter of both is not updated synchronously. Then S and T initiate a new authenticated communication round, the other party cannot successfully authenticate the session message generated by one party because their key parameters lose synchronization. In this case, S and T initiate a new authentication session using the pre-stored (SID, K_{em}) and (C_{em}, R_{em}). Specifically, T randomly selects one of the unused pairs of $(sid_x, k_{emx}) \in (SID, K_{em})$ from the pre-stored (SID, K_{em}), and similarly, S randomly selects a pair of $CRPs \in (C_{em}, R_{em})$ from the pre-stored (C_{em}, R_{em}) for authentication communication. Once each time a pair is $(sid_x, k_{emx}) \in (SID, K_{em})$ or $CRPs \in (C_{em}, R_{em})$ used, it needs to be deleted from both S and T respectively, which is important because no update mechanism is provided for contingency pairs. Obviously, according to the (SID, K_{em}) and (C_{em}, R_{em}) pre-stored by T and S respectively, so the communication between S and T can only resist at most n desynchronization attacks. Although Gope states that their scheme can resist desynchronization attacks, our analysis shows that the scheme cannot resist n consecutive desynchronization attacks. Specifically, T appends a "Re-Load" message to M_1 in the $i + n$ authentication session round after it has suffered $n - 1$ desynchronization attacks. When S receives the "Re-Load" message, it generates a new set of pairs and sends them to T with session key K_i encryption. According to the protocol authentication procedure, the session key $K_i = \mathsf{FE.Rec}(R_i, hd_i)$ is reconstructed by S after receiving M_3. However, the adversary initiates a desynchronization attack that precisely blocks M_3, hence S cannot reconstruct the K_i shared with T. Therefore, the scheme cannot resist the desynchronization attack.

B Physical Unclonable Functions

According to the number of Challenge-Response Pairs ($CRPs$) generated, PUF can be divided into strong PUF and weak PUF types. Since only a limited number of $CRPs$ can be generated for weak PUFs, once an adversary gains access to $CRPs$, the adversary can easily clone all $CRPs$, which is impossible with strong PUFs. The static random access memory PUF (SRAM PUF) is classified as weak because only a limited number of $CRPs$ can be generated. In contrast, traditional PUFs with large $CRPs$ (such as arbitrator PUFs) are classified as strong PUFs, which are commonly used in authentication protocols [26–28]. The PUF structure selected for placement in the tag is the SRAM PUF in the Gope et al. scheme. However, Talukder et al. point out that memory-based PUFs are vulnerable as well in [28].

Configurable Tristate Physical Unclonable Function (CT PUF). Zhang et al. [29] proposed configurable tristate PUF (CT PUF) to solve the vulnerability of PUF against the machine learning modeling attack. To improve its security, the CT PUF use the bit-wise XOR obfuscation mechanism to hide the

mapping relationship between challenge and response pairs. The XOR bit-wise obfuscation mechanism consists of two phases, the Preparation phase and the Obfuscation phase. The detailed as depicted in Fig. 5.

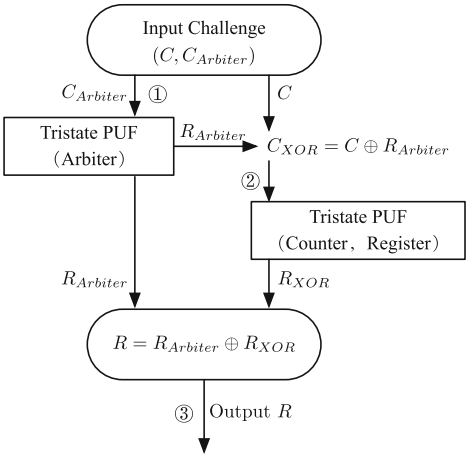

Fig. 5. The bitwise XOR obfuscation mechanism [29].

C Feldman's (t, n)-threshold Verifiable Secret Sharing Scheme

Feldman proposed a non-interactive scheme for achieving verifiability in Shamir's threshold secret sharing scheme in [30]. The following three algorithms are included.

– VSS.Share($\cdot$): For a secret s. First, the dealer selects a random polynomial $(a_0 = s)$:

$$F(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1}, a_0, a_1, \cdots, a_{t-1} \in_R \mathbb{Z}_n.$$

Next, the dealer distributes $s_i = F(i), i = 1, 2, \cdots, m$ to the P_i over a secure channel. Meanwhile, the dealer broadcasts the commitments $\alpha = \{\alpha_j = g^{a_j}, 0 \leq j \leq t-1\}$.

– VSS.Verify($\cdot$): Then, once P_i receives s_i, the validity of the each share can be verified by the Eq. (1).

$$g^{s_j} = \prod_{i=0}^{t-1} \alpha_i^{j^i}. \tag{1}$$

– $\mathsf{VSS.Rec}(\cdot)$: t parties can reconstruct the secret. The parties first check the validity of the received share s_i by comparing the Eq. (1). If it holds, then the s can be reconstruct with Lagrange Interpolation (the Eq. (2)).

$$P(0) = \sum_{i=0}^{t-1} P(x_i) \prod_{\substack{j=0 \\ j \neq i}}^{t-1} \frac{(0 - x_j)}{x_i - x_j} = s. \tag{2}$$

D Basic Notations and Logical Postulates of BAN Logical

D.1 Basic Notations

The basic notations and descriptions of BAN logic are listed in Table 7.

Table 7. Basic notations of BAN logical

Notation	Description	Notation	Description
$P \mid\equiv X$	P believes X	$P \triangleleft X$	P sees X
$P \mid\sim X$	P once said X	$P \mid\Rightarrow X$	P has jurisdiction over X
$\#(X)$	X is freshness	$P \overset{K}{\longleftrightarrow} Q$	P and Q share the K
$\{X\}_K$	X encrypted under the K	$\langle X \rangle_K$	X hasded by the K

D.2 Logical Postulates

The logical postulates and descriptions of BAN logic are shown as follows.

Lemma 1 ***(R1:The message-meaning rules).*** *If P believes that the key K is shared with Q and receives a message X encrypted by K: $\{X\}_K$, then P believes that Q once said the message X.*

$$\frac{P \mid\equiv P \overset{K}{\longleftrightarrow} Q, P \triangleleft \{X\}_K}{P \mid\equiv Q \mid\sim X} \tag{R1}$$

Lemma 2 ***(R2:The jurisdiction rule).*** *If P believes that X could have been uttered only recently and that Q once send message X, then P believed that Q believes the message X.*

$$\frac{P \mid\equiv Q \mid\Rightarrow X, P \mid\equiv Q \mid\equiv X}{P \mid\equiv X} \tag{R2}$$

Lemma 3 ***(R3: The nonce-verification rule).*** *If P believes that X could have been uttered only recently (in the present) and that Q once said X (either in the past or in the present), then P believes that Q believes X.*

$$\frac{P \mid\equiv \#(X), P \mid\equiv Q \mid\sim X}{P \mid\equiv Q \mid\equiv X} \tag{R3}$$

Lemma 4 ***(R4: The belief rule).*** *If P believes that Q sends the tuple $\{X, Y\}$, then P believes message X was sent by Q.*

$$\frac{P \triangleleft \{X, Y\}}{P \triangleleft X} \tag{R4}$$

Lemma 5 ***(R5: The freshness rule).*** *If P believes that part of the X is fresh, then P believes that the entire $\#(X, Y)$ must also be fresh.*

$$\frac{P \mid\equiv \#(X)}{P \mid\equiv \#(X, Y)} \tag{R5}$$

E Security Simulation Using Scyther

In this subsection, we evaluate the security level of our protocols using Scyther, a widely accepted tool for analyzing and verifying the security of protocols. The tool provides a user-friendly graphical user interface to facilitate the analysis and verification of complex attack scenarios on protocols.

The Security Protocol Description Language (SPDL) description for function and messages, the SPDL description for role S and the SPDL description for role T as follows. When the protocol model is established, two roles S and R are defined, representing the communication parties server and tag, respectively. In addition, a secret declaration for each message sent and received by the two communication parties. Figure 6 shows the verification result of Scyther simulation, which shows that no attack against the proposed protocol can be detected.

SPDL description for function and messages

```
1  //PUF–Based RFID authentication scheme
2  hashfunction H;                /*Secure Hash function*/
3  const XOR      : Function; /*Bitwise XOR function*/
4  const CTPUF    : Function; /*PUF function */
5  const MTCTPUF : Function; /*Soft Model function */
6  const CON      : Function; /* Conjunction function*/
7  macro CT        = XOR(CArbiter, NT);
8  macro NTstar    = XOR(NT, KS);
9  macro M1        = CON(SID, CT, NTstar);
10 macro NT'       = XOR(NTstar,KS);
11 macro CArbiter'= XOR(CT,NT);
```

```
macro NSstar    = XOR(KT,NS);
macro ResS      = H(Ci,NT',KT,NSstar);
macro M2        = CON(Ci,NSstar,ResS);
macro ResS'     = H(Ci,NT,KT,NSstar);
macro RArbiter  = CTPUF(CArbiter);
macro CXOR      = XOR(RArbiter,Ci);
macro RXOR      = CTPUF(CXOR);
macro R         = XOR(RArbiter,RXOR);
macro NS'       = XOR(KT,NSstar);
macro RT        = XOR(R,NS');
macro STX       = XOR(S1x,KT);
macro STY       = XOR(S1y,NS');
macro ResT      = H(NS',KT,RT,STX,STY);
macro M3        = CON(RT, STX, STY, ResT);
macro ResT'     = H(NS,KT,RT,STX,STY);
macro R'        = XOR(RT,NS);
macro S1x'      = XOR(STX,KT);
macro S1y'      = XOR(STY,NS);
macro RArbiter'= VSSRec(S1,S2);
macro CXOR'     = XOR(RArbiter' ,Ci);
macro RXOR'     = CTPUF(CXOR');
macro RHat      = XOR(RArbiter', RArbiter');
macro RS        = XOR(R, NT');
macro M4        = CON(RS);
macro R'        = XOR(RS, NT);
```

SPDL description for roles S

```
role T{
var   NS : Nonce;
var   Ci : Nonce;
fresh NT :Nonce;
const KT,KS,CArbiter,RArbiter,CXOR,RXOR,
R,S1x,S1y,SID,RS,ResS  : Ticket;
send_1(T,S, M1);
recv_2(S,T,M2);
match(ResS,ResS');
send_3(T,S,M3);
recv_4(S,T,M4);
claim(T,Secret,CT);
claim(T,Secret,NTstar);
claim(T,Secret,RArbiter);
claim(T,Secret,R);
claim(T,Secret,RT);
claim(T,Secret,STX);
```

```
18 claim(T,Secret,STY);
19 claim(T,Secret,ResT);
20 claim(T,Secret,NT);
21 claim(T,Niagree);
22 claim(T,Nisynch);
23 claim(T,Alive);
24 claim(T,Weakagree);
25 }
```

SPDL description for roles R

```
1  role S{
2  var   NT : Nonce;
3  fresh NS : Nonce;
4  fresh Ci : Nonce;
5  const KT,KS,CArbiter,RArbiter,CXOR,RXOR,R,
6  S1x,S1y,SID,RS,ResS : Ticket;
7  recv_1(T,S, M1);
8  send_2(S,T,M2);
9  recv_3(T,S,M3);
10 match(ResT,ResT');
11 send_4(S,T,M4);
12 claim(S,Secret,NSstar);
13 claim(S,Secret,ResS);
14 claim(S,Secret,RS);
15 claim(S,Secret,NS);
16 claim(S,Niagree);
17 claim(S,Nisynch);
18 claim(S,Alive);
19 claim(S,Weakagree);
20 }
```

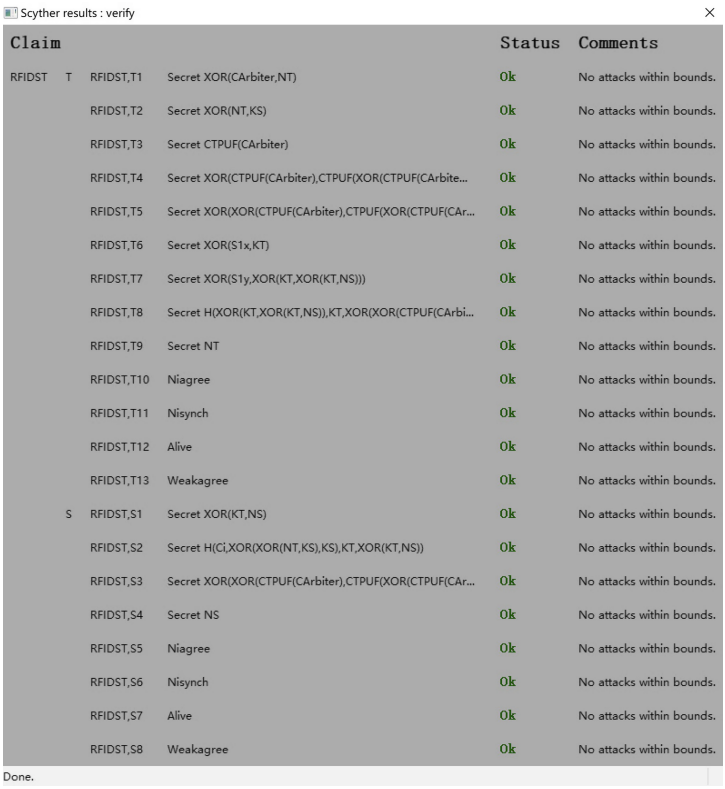

Fig. 6. Verification result of Scyther simulation.

References

1. Huang, Y.J., Yuan, C.C., Chen, M.K., et al.: Hardware implementation of RFID mutual authentication protocol. IEEE Trans. Ind. Electron. **57**(5), 1573–1582 (2009)
2. Aghili, S.F., Mala, H., Schindelhauer, C., et al.: Closed-loop and open-loop authentication protocols for blockchain-based IoT systems. Inf. Process. Manag. **58**(4), 102568 (2021)
3. Salem, F.M., Amin, R.: A privacy-preserving RFID authentication protocol based on El-Gamal cryptosystem for secure TMIS. Inf. Sci. **527**, 382–393 (2020)
4. Agrahari, A.K., Varma, S.: A provably secure RFID authentication protocol based on ECQV for the medical internet of things. Peer-to-Peer Netw. Appl. **14**(3), 1277–1289 (2021)
5. Kumar, V., Ahmad, M., Mishra, D., et al.: RSEAP: RFID based secure and efficient authentication protocol for vehicular cloud computing. Veh. Commun. **22**, 100213 (2020)
6. Gope, P., Millwood, O., Saxena, N.: A provably secure authentication scheme for RFID-enabled UAV applications. Comput. Commun. **166**, 19–25 (2021)

7. Park, H., Roh, H., Lee, W.: Tagora: a collision-exploitative RFID authentication protocol based on cross-layer approach. IEEE Internet Things J. **7**(4), 3571–3585 (2020)
8. Gao, L., Ma, M., Shu, Y., et al.: An ultralightweight RFID authentication protocol with CRC and permutation. J. Netw. Comput. Appl. **41**, 37–46 (2014)
9. Gope, P., Lee, J., Quek, T.Q.S.: Lightweight and practical anonymous authentication protocol for RFID systems using physically unclonable functions. IEEE Trans. Inf. Forensics Secur. **13**(11), 2831–2843 (2018)
10. Peris-Lopez, P., Hernandez-Castro, J C., Estévez-Tapiador, J M., et al.: LMAP: a real lightweight mutual authentication protocol for low-cost RFID tags. In: CONFERENCE 2006, the 2nd Workshop on RFID Security, vol 6, pp. 1–12. Springer, Heidelberg (2006)
11. Peris-Lopez, P., Hernandez-Castro, J.C., Estevez-Tapiador, J.M., Ribagorda, A.: EMAP: an efficient mutual-authentication protocol for low-cost RFID tags. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM 2006. LNCS, vol. 4277, pp. 352–361. Springer, Heidelberg (2006). https://doi.org/10.1007/11915034_59
12. Peris-Lopez, P., Hernandez-Castro, J.C., Estevez-Tapiador, J.M., Ribagorda, A.: M^2AP: a minimalist mutual-authentication protocol for low-cost RFID tags. In: Ma, J., Jin, H., Yang, L.T., Tsai, J.J.-P. (eds.) UIC 2006. LNCS, vol. 4159, pp. 912–923. Springer, Heidelberg (2006). https://doi.org/10.1007/11833529_93
13. Chien, H.Y., Huang, C.W.: Security of ultra-lightweight RFID authentication protocols and its improvements. ACM SIGOPS Oper. Syst. Rev. **41**(4), 83–86 (2007)
14. Li, T., Wang, G.: Security analysis of two ultra-lightweight RFID authentication protocols. In: Venter, H., Eloff, M., Labuschagne, L., Eloff, J., von Solms, R. (eds.) SEC 2007. IIFIP, vol. 232, pp. 109–120. Springer, Boston, MA (2007). https://doi.org/10.1007/978-0-387-72367-9_10
15. Chien, H.Y.: SASI: a new ultralightweight RFID authentication protocol providing strong authentication and strong integrity. IEEE Trans. Dependable Secur. Comput. **4**(4), 337–340 (2007)
16. Cao, T., Bertino, E., Lei, H.: Security analysis of the SASI protocol. IEEE Trans. Dependable Secur. Comput. **6**(1), 73–77 (2008)
17. Sun, H.M., Ting, W.C., Wang, K.H.: On the security of Chien's ultralightweight RFID authentication protocol. IEEE Trans. Dependable Secur. Comput. **8**(2), 315–317 (2009)
18. Tian, Y., Chen, G., Li, J.: A new ultralightweight RFID authentication protocol with permutation. IEEE Commun. Lett. **16**(5), 702–705 (2012)
19. Li, W., Xiao, M., Li, Y., Mei, Y., Zhong, X., Tu, J.: Formal analysis and verification for an ultralightweight authentication protocol RAPP of RFID. In: Du, D., Li, L., Zhu, E., He, K. (eds.) NCTCS 2017. CCIS, vol. 768, pp. 119–132. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-6893-5_9
20. Wang, S H., Han, Z J., Liu, S J., et al.: Security analysis of RAPP an RFID authentication protocol based on permutation. College of computer, Nanjing University of Posts and Telecommunications, pp. 293–308 (2012). (in Chinese)
21. Pappu, R., Recht, B., Taylor, J., et al.: Physical one-way functions. Science **297**(5589), 2026–2030 (2002)
22. Li, T., Liu, Y.L.: A double PUF-based RFID authentication protocol. J. Comput. Res. Dev. **58**(8), 1801 (2021). (in Chinese)
23. Akgün, M., Çağlayan, M.U.: Providing destructive privacy and scalability in RFID systems using PUFs. Ad Hoc Netw. **32**, 32–42 (2015)

24. Liang, W., Xie, S., Long, J., et al.: A double PUF-based RFID identity authentication protocol in service-centric internet of things environments. Inf. Sci. **503**, 129–147 (2019)
25. Nimmy, K., Sankaran, S., Achuthan, K.: A novel lightweight PUF based authentication protocol for IoT without explicit CRPs in verifier database. J. Ambient Intell. Humaniz. Comput. 1–16 (2021). (Published online)
26. Herder, C., Yu, M.D., Koushanfar, F., et al.: Physical unclonable functions and applications: a tutorial. Proc. IEEE **102**(8), 1126–1141 (2014)
27. Rührmair, U., Sölter, J., Sehnke, F., et al.: PUF modeling attacks on simulated and silicon data. IEEE Trans. Inf. Forensics Secur. **8**(11), 1876–1891 (2013)
28. Talukder, B.M.S.B., Ferdaus, F., Rahman, M.T.: Memory-based PUFs are vulnerable as well: a non-invasive attack against SRAM PUFs. IEEE Trans. Inf. Forensics Secur. **16**, 4035–4049 (2021)
29. Zhang, J., Shen, C., Guo, Z., et al.: CT PUF: configurable tristate puf against machine learning attacks for IoT security. IEEE Internet Things J. **9**(16), 14452–14462 (2022)
30. Feldman, P.: A practical scheme for non-interactive verifiable secret sharing. In: 28th Annual Symposium on Foundations of Computer Science FOCS 1987, pp. 427–438. IEEE, Los Angeles, CA, USA (1987)
31. Burrows, M., Abadi, M., Needham, R M.: A logic of authentication. Proc. R. Soc. Lond. A Math. Phys. Sci. **426**(1871), 233–271 (1989)