Journal of Information Security and Applications 67 (2022) 103194

Contents lists available at ScienceDirect

Journal of Information Security and Applications

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

Improving RFID/IoT-based generalized ultra-lightweight mutual authentication protocols

Masoumeh Safkhani [a,b], Samad Rostampour [c,d], Ygal Bendavid [c,*], Sadegh Sadeghi [e,f], Nasour Bagheri [g,b]

[a] Computer Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Postal code: 16788-15811, Iran
[b] School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran 19538-33511, Iran
[c] Department of Analytics Operations and Technology, School of Management, Universite du Quebec a Montreal (UQAM), Montreal, QC., H2X 3X2, Canada
[d] Department of Computer Science, Vanier College, Montreal, QC, Canada
[e] Department of Mathematics, Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan 45137-66731, Iran
[f] Research Center for Basic Sciences and Modern Technologies (RBST), Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan 45137-66731, Iran
[g] CPS^2 Lab, Electrical Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Postal code: 16788-15811, Iran

ARTICLE INFO

Keywords:
Internet of things
Authentication
RFID
Security
Ultra-lightweight

ABSTRACT

With the increase of connected devices being introduced into the market each day, the Internet of Things (IoT) vision is progressively becoming a reality. This phenomenon however increases key security risks flagged by the scientific and research community and industry professionals. Devices featuring limited security capabilities are of particular concern in IoT adoption, including passive Radio Frequency Identification (RFID) tags. In response to such security limitations, several ultra-lightweight authentication protocols have been proposed, although, most of them exhibit various vulnerabilities. In this study, we evaluate the security level of recent ultra-lightweight mutual authentication protocols and show their susceptibility to replay and desynchronization attacks. Through this research, we also show that these protocols can be grouped into a generalized version of ultra-lightweight mutual authentication protocols (GUMAPs) and classify them into two categories: (i) GUMAP1, where both parties (tag and reader) maintain a history of old parameters; and (ii) GUMAP2, where only one party maintains a history of old parameters. We then establish that both groups are vulnerable to replay and desynchronization attacks. To eliminate these vulnerabilities, we propose a more secure generalized improved mutual authentication protocol (GIMAP). To address the security issues, we present a new message authentication code (MAC) function for GIMAP and prove that the new protocol can satisfy the security requirement involved in lightweight protocols. The security analysis of GIMAP is also supported by the formal security analysis, using two widely accepted approaches, namely BAN logic and the Scyther tool.

1. Introduction

The underlying vision of the Internet of things (IoT) is that any object can be equipped with technology to become a computing device, interacting autonomously and in real-time with its environment. This vision is becoming a reality as the worldwide number of IoT-connected devices is increasing exponentially. Business Fortune Insights (BFI) released a study suggesting that the global IoT market – valued at US$ 250.7 billion in 2019 – is projected to reach US$1,463.6 billion by 2027, at a CAGR of 24.9% during the forecast period of 2020–2027; pointing out the importance of security in the Software and Services offering [1].

More precisely, a recent report from Data Bridge Market Research [2] estimates that the global IoT security market is set to witness a CAGR of 34% in the forecast period of 2019 to 2026. Considering the current state of the IoT threat landscape, analysts and industry professionals agree that cybersecurity is a strategic imperative for every corporate executive involved in an IoT project [3] and urge CSOs to act immediately to "reduce their organizations' exposure to IoT-initiated attacks" [4].

With billions of connected heterogeneous devices [5] being developed and deployed using different IoT frameworks [6], understanding how to build low-cost/high security IoT devices becomes critical.

* Corresponding author.
E-mail addresses: Safkhani@sru.ac.ir (M. Safkhani), rostamps@vaniercollege.qc.ca (S. Rostampour), bendavid.ygal@uqam.ca (Y. Bendavid), s.sadeghi@iasbs.ac.ir (S. Sadeghi), Nbagheri@sru.ac.ir (N. Bagheri).

https://doi.org/10.1016/j.jisa.2022.103194

Given the ubiquitousness of IoT device-related security issues (e.g. authentication, authorization, integrity, confidentiality, non-repudiation, availability, and privacy), the scientific and research community is also calling for security concerns and requirements for all layers of any IoT-based infrastructure [7–9]. This is particularly important with low-cost connecting devices featuring limited security capabilities, such as passive radio frequency identification (RFID) transponders (also known as tags). More specifically, passive Ultra High Frequency (UHF) RFID tags are increasingly used in various sectors such as retail, logistics, health care/hospitals, event management, etc. Although the use of these high-performance, low-price tags is increasingly justified from a business perspective, it represents a drawback from a security perspective. Indeed, passive RFID UHF tags are equipped with a highly constrained microchip holding limited processing power and memory capabilities. For instance: (i) the integrated circuit (IC) of these tags features a limited number of logical gates (gate equivalents-GEs) referring to a tag's weight; and (ii) they use a 16-bit cyclic redundancy check (CRC) to ensure the validity of certain authentication commands [10].

2. Related works

In response to such security limits, several lightweight and ultra-lightweight RFID authentication protocols have been proposed in recent years [11–13]. However, most of these protocols exhibit various vulnerabilities such as desynchronization, traceability, replay, and secret disclosure attacks [14,15]. Therefore, with a restriction on the number of gates on a tag's IC, reducing the opportunities for designing and implementing a secure protocol, a common approach to designing an ultra-lightweight mutual authentication protocol (UMAP) must rely on simple bitwise operations rather than complicated functions. This option, however, opens up security breaches in the protocol. Additionally, because the wireless communication channel between the reader and tag, a third-party (adversary) could sniff and eavesdrop the transferred packets, manipulate them and compromise the security level of the authentication protocol. Considering these security issues, researchers such as Tian et al. have proposed an ultra-lightweight mutual RFID authentication protocol with permutation (RAPP) [16]. Although RAPP remains a vulnerable protocol [17–19], other researchers leveraged it to develop improved protocols such as an RFID authentication protocol for low-cost tags (R2AP) [20], an RFID authentication protocol using a recursive hash (RCIA) [21], an RFID pseudo-Kasami code-based mutual authentication protocol (KMAP) [22], a succinct and lightweight authentication protocol (SLAP) [23], a strong authentication and strong integrity (SASI), a UMAP using recursive hash function (SASI+) [24], and UMAPs based on secret sharing (UMAPSS) [25]. With regard to RAPP, all the above mentioned protocols use three simple operations to provide the desired complexity: (i) bitwise XOR, (ii) left rotation, and (iii) a very lightweight nonlinear function. However, the general framework of these protocols mostly follows the initial framework proposed by Tian et al. with a slight difference that can divide these protocols into two categories: a category in which only one party keeps the history of the old data, and another category for which both parties keep it.

2.1. Our contributions

In order to contribute to research and practice-driven applications with relevance to information security and applications the main goal of the paper is to propose a more secure RFID/IoT generalized ultra-lightweight mutual authentication protocol. More specifically:

(1) We evaluate the security of the protocol proposed by Liu et al. namely UMAPSS [25], as the last version of this RAPP-based series, and we show that the protocol is vulnerable to efficient attacks. More precisely, we show that any adversary who eavesdrops on a session of the protocol can later apply it to the tag and will be authenticated with the probability of '1'. We also extend the attack to a desynchronization attack, where the adversary can desynchronize any tag only on the strength of the complexity of storing three sessions of the protocol.

(2) We generalize the proposed attacks and apply them to the recent related protocols. More precisely, we introduce two categories of protocols, where the overall structure of each category is almost identical. We define a Generalized version of RAPP-based ultra-lightweight mutual authentication protocols (GUMAPs) that can be divided in two groups: (i) GUMAP1, where both the parties maintain a history of old parameters, e.g. RCIA, KMAP, SLAP, and SASI$^+$, and (ii) GUMAP2, where only one party maintains it, e.g. RAPP, R^{2}AP, and UMAPSS (Fig. 1). It should be noted that there is no constraint on the length of any of the protocols' parameters. For example, in the case of SLAP, wherein a tag has two keys, in our generalized description, we concatenate the keys as one key. We show that any protocol in GUMAP1 or GUMAP2 can be desynchronized with the complexity of 5 and 3 sessions of the protocol respectively. In addition, we prove that all mentioned protocols are vulnerable to the replay attack. The complexity of the replay attack in both cases is just eavesdropping the transferred messages in a session and later broadcasting them.

(3) In order to solve the reported security issues, we propose a new permutation-based message authentication code (MAC) function and utilize the MAC function in our enhanced protocol as a generalized improved mutual authentication protocol (GIMAP) that addresses the common shortcoming of the above mentioned previous protocols (i.e. ensuring it is secure against the aforementioned attacks) (Fig. 1). Beyond informal security reasoning, we evaluate the security of GIMAP formally using BAN logic and the Scyther tool. In addition, we provide the efficiency evaluation of GIMAP in the comparison of other protocols.

2.2. Paper organization

In the next section (Section 3), we introduce the notations and preliminaries. In Section 4, we present the security analysis of UMAPSS, as the latest version of the RAPP-based protocol. In section Section 5, we introduce generalized versions of ultra-lightweight mutual authentication protocols, where two parties maintain a history of old parameters (GUMAP1) or only one party (GUMAP2). Generalized replay and desynchronization attacks against GUMAP1 and GUMAP2 are also presented in this section. In Section 6, we propose a new message authentication code (MAC) algorithm that is used in our proposed scheme. In Section 7, we propose a generalized improved mutual authentication protocol (GIMAP) that addresses the common weaknesses mentioned in the previous protocols (i.e. secure against the presented attacks). The informal and formal security evaluations of GIMAP are represented in section Section 8. Then in Section 9 we present a performance and comparison of GIMAP and compare it with other protocols. Finally, this is followed by the conclusion in section Section 10.

3. Preliminaries

3.1. Notation

Table 1 lists the notations and describes each symbol. Given bit strings x and y, we denote their bitwise XOR by $x \oplus y$ and their bitwise AND by $x \wedge y$. In this study, $Rot(x; y)$ is assigned to bitwise right/ left rotation of x as a factor of y, and $\lll$ only shows the bitwise left rotation. Here, $wt(y)$ is the Hamming weight of string y (amount of bit(s) in y that is equal to one).

However, in this paper we evaluate seven different protocols, in order to summarize the descriptions of the protocols we only describe UMAPSS as the last version of RAPP-based protocols in this section and the descriptions and vulnerabilities of other protocols have been explained one by one in [26].

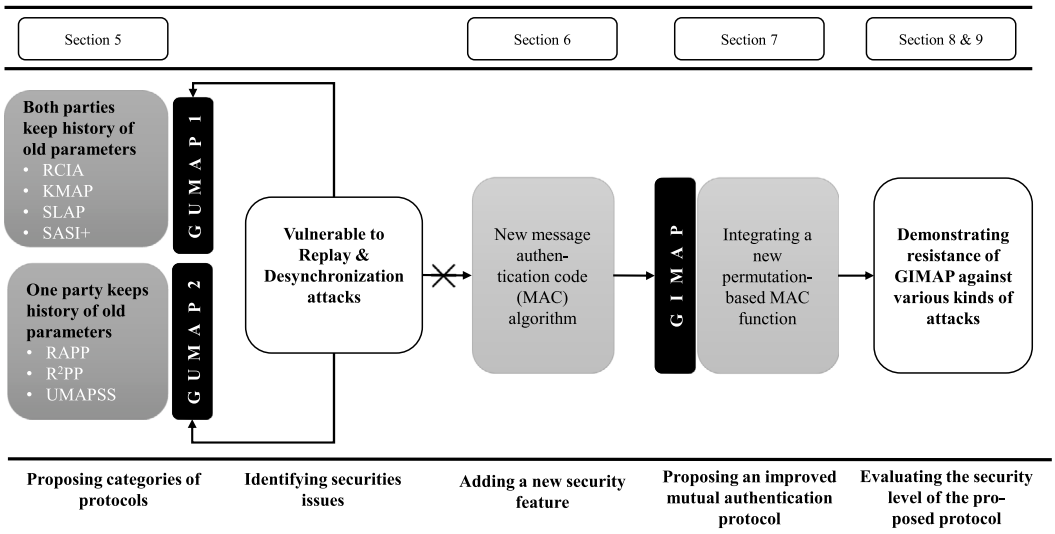

Fig. 1. Framework of the article.

Table 1
List of notations.

Symbol	Description
R	An RFID reader
TD	A Trusted Database
T	An RFID tag
$Rot(x, y)$	Left/Right rotation of x according to y.
$\lll$	The left rotation operation
X_L	Left half of X
X_h	Right half of X
ID	Tag's ID
IDS	Tag's Pseudonym ID
K^i	A Secret Key
$R_h(x)$	A Recursive hash function
$per(x, y)$	A Permutation function based on x and y
n^i	A random number.
$\mathcal{A}$	An adversary
$\wedge$	Bitwise AND operation
$PRNG(.)$	The pseudo random number generator
$\oplus$	The bitwise XOR operation
$\|\|$	The concatenation operation
$\%$	The modular operation
$\overline{x}$	The bitwise not of x

3.2. Evaluating UMAPSS security

UMAPSS [25] is a protocol based on three entities: the tag (T), reader (R), and trusted database (TD). The main functions in the tag are rotation, $PRNG$ and addition. Also, there is a key that consists of the concatenation of three parts ($K = K^1 \| K^2 \| K^3$). The authentication phase of UMAPSS protocol is depicted in Fig. 2 and works as follows:

1. The reader starts the session and sends a Hello message.
2. After receiving Hello message, the tag sends its IDS to the reader.
3. The reader generates μ as a random number, calculates A and B and transfers them to the tag.
   $A = Rot(Rot(IDS + \mu, K^1), K^2) + K^3$
   $B = Rot(Rot(IDS + K^1, \mu), K^3) + K^2$
4. The tag extracts the random number by A and verifies B. If B is correct, the tag authenticates the reader.
5. Therefore, the tag computes C and D and sends them to the reader. It also updates IDS, K^1 and K^2.
   $C = Rot(K^1 + K^2, K^3 + \mu) + x_r$
   $D = Rot(K^2 + K^3, K^1 + \mu) + y_r$
   If the reader receives C and D in the acceptable time period, it extracts x_r and y_r to authenticates the tag. If the received values are correct, it verifies the tag and updates IDS, K^1 and K^2 as follows:
   $IDS^{old} = IDS$
   $K_i^{old} = K^i$
   $IDS^{new} = Rot(IDS^{old} + \mu + x_r, K_1^{old} + K_2^{old} + y_r) + K_3^{old}$
   $K_1^{new} = Rot(K_2^{old} + \mu, K_3^{old} + x_r) + K_1^{old}$
   $K_2^{new} = Rot(K_3^{old} + \mu, K_1^{old} + x_r) + K_2^{old}$
   $K_3^{new} = Rot(K_1^{old} + \mu, K_2^{old} + x_r) + K_3^{old}$

For more details of UMAPSS, we refer to [25].

4. Desynchronization and reply attack on UMAPSS

In this section, we present a detailed desynchronization attack against UMAPSS [25]; it is a special case of GUMAP2 (a simplified version). In UMAPSS, only the reader maintains a record of old and new shared parameters. The proposed desynchronization attack on UMAPSS operates as follows:

1. **In session** i:
   Assuming a legitimate reader R communicates with a legitimate tag T,
   (a) R receives IDS from T, generates μ, and sends $A = Rot(Rot(IDS + \mu, K^1), K^2) + K^3$ and $B = Rot(Rot(IDS + K^1, \mu), K^3) + K^2$ to T.
   (b) T authenticates R, calculates and sends $C = Rot(K^1 + K^2, K^3 + \mu) + x_r$ and $D = Rot(K^2 + K^3, K^1 + \mu) + y_r$ to R_j. T also updates the sub-keys and IDS as follows:

$$IDS^{old} = IDS;$$

$$K_i^{old} = K_i;$$

$$IDS^{new} = Rot(IDS^{old} + \mu + x_r, K_1^{old} + K_2^{old} + y_r) + K_3^{old};$$

$$K_1^{new} = Rot(K_2^{old} + \mu, K_3^{old} + x_r) + K_1^{old};$$

$$K_2^{new} = Rot(K_3^{old} + \mu, K_1^{old} + x_r) + K_2^{old};$$

$$K_3^{new} = Rot(K_1^{old} + \mu, K_2^{old} + x_r) + K_3^{old}.$$

Reader R_j		**Tag** T_r
$(IDS, K, \mu, (x_j, y_j), K_1^{new}, K_2^{new}, ID)$		$(IDS^{old}, K^{old}, (x_r, y_r), IDS^{new}, K^{new}, ID)$
	$\xrightarrow{Hello}$	
Generates μ	$\xleftarrow{IDS}$	
$A = Rot(Rot(IDS + \mu, K^1), K^2) + K^3$		
$B = Rot(Rot(IDS + K^1, \mu), K^3) + K^2$		Extracts μ from A
	$\xrightarrow{A,B}$	Verifies B to authenticate R_j
		$C = Rot(K^1 + K^2, K^3 + \mu) + x_r$,
	$\xleftarrow{C,D}$	$D = Rot(K^2 + K^3, K^1 + \mu) + y_r$
If received C and D within the expected time ΔT, extracts x_r and y_r from C and D. verifies whether (x_r, y_r) is a valid share to authenticate T_r.		Updates IDS, K^1, K^2
Updates IDS, K^1, K^2		
$IDS^{old} = IDS$, $K_i^{old} = K^i$, $IDS^{new} = Rot(IDS^{old} + \mu + x_r, K_1^{old} + K_2^{old} + y_r) + K_3^{old}$, $K_1^{new} = Rot(K_2^{old} + \mu, K_3^{old} + x_r) + K_1^{old}$, $K_2^{new} = Rot(K_3^{old} + \mu, K_1^{old} + x_r) + K_2^{old}$, $K_3^{new} = Rot(K_1^{old} + \mu, K_2^{old} + x_r) + K_3^{old}$		

Fig. 2. Mutual authentication phase of UMAPSS [25].

(c) R authenticates T and updates the sub-keys and IDS as follows:

$$IDS = Rot(IDS + \mu + x_r, K_1 + K_2 + y_r) + K_3;$$

$$K_1 = Rot(K_2 + \mu, K_3 + x_r) + K_1;$$

$$K_2 = Rot(K_3 + \mu, K_1 + x_r) + K_2;$$

$$K_3 = Rot(K_1 + \mu, K_2 + x_r) + K_3.$$

(d) The adversary $\mathcal{A}$ eavesdrops IDS, A, B, C and D and stores them. Furthermore, $\mathcal{A}$ blocks C and D. Hence, the reader's record remains (IDS^{old}, K^{old}).

2. In session $i + 1$:

(a) T sends IDS^{new} but is not authenticated by the reader. Hence, it sends $IDS = IDS^{old}$ to R. R generates μ' and sends $A' = Rot(Rot(IDS + \mu', K_1), K_2) + K_3$ and $B' = Rot(Rot(IDS + K_1, \mu'), K_3) + K_2$ to T_r.

(b) T authenticates R, calculates and sends $C = Rot(K_1 + K_2, K_3 + \mu') + x_r$ and $D = Rot(K_2 + K_3, K1 + \mu') + y_r$ to R_j. T_r also updates the sub-keys and IDS as follows:

$$IDS^{old} = IDS;$$

$$K_i^{old} = K_i;$$

$$IDS'^{new} = Rot(IDS^{old} + \mu' + x_r, K_1^{old} + K_2^{old} + y_r)$$

$$+K_3^{old};$$

$$K_1'^{new} = Rot(K_2^{old} + \mu', K_3^{old} + x_r) + K_1^{old};$$

$$K_2'^{new} = Rot(K_3^{old} + \mu', K_1^{old} + x_r) + K_2^{old};$$

$$K_3'^{new} = Rot(K_1^{old} + \mu', K_2^{old} + x_r) + K_3^{old}.$$

(c) R authenticates T and updates the sub-keys and IDS as follows:

$$IDS' = Rot(IDS + \mu' + x_r, K_1 + K_2 + y_r) + K_3;$$

$$K_1' = Rot(K_2 + \mu', K_3 + x_r) + K_1;$$

$$K_2' = Rot(K_3 + \mu', K_1 + x_r) + K_2;$$

$$K_3' = Rot(K_1 + \mu', K_2 + x_r) + K_3.$$

3. In session $i + 2$:

(a) $\mathcal{A}$ impersonates R, sends $Hello$ to T and T sends IDS^{new}. However, $\mathcal{A}$ sends another $Hello$ to T and T replies with IDS^{old}.

(b) $\mathcal{A}$ sends the eavesdropped $A = Rot(Rot(IDS+\mu, K_1), K_2)+K_3$ and $B = Rot(Rot(IDS + K_1, \mu), K_3) + K_2$ to T.

(c) T authenticates $\mathcal{A}$ as a legitimate reader and calculates and sends $C = Rot(K_1 + K_2, K_3 + \mu) + x_r$ and $D = Rot(K_2 + K_3, K1+\mu)+y_r$ to it. T also updates the sub-keys and IDS as follows:

$$IDS^{old} = IDS;$$

$$K_i^{old} = K_i;$$

$$IDS^{new} = Rot(IDS^{old} + \mu + x_r, K_1^{old} + K_2^{old} + y_r)$$

$$+K_3^{old};$$

$$K_1^{new} = Rot(K_2^{old} + \mu, K_3^{old} + x_r) + K_1^{old};$$

$$K_2^{new} = Rot(K_3^{old} + \mu, K_1^{old} + x_r) + K_2^{old};$$

$$K_3^{new} = Rot(K_1^{old} + \mu, K_2^{old} + x_r) + K_3^{old}.$$

At the end of this attack, the tag's IDS and K for the old and new versions are IDS^{old}, K^{old}, IDS^{new}, and K^{new}, respectively; meanwhile, the reader's shared values are IDS' and K'. Therefore, it is evident that none of the tag's records is equal to the reader's record. Hence, the reader and tag will be desynchronized after the above attack with a probability of approximately one. Given that the reader and tag have been desynchronized, the tag will not update its secrets. Hence, the adversary can conveniently trace the tag holder or impersonate the reader to the tag at any time.

It is clear that Steps 1d, 3b and 3c of the above mentioned attack can be considered a replay attack on UMAPSS, for which the complexity is simply eavesdropping a session of the protocol between a legitimate reader and the target tag T -as explained in Step 1d- and another session later, to reply the eavesdropped values to T, similar to Steps 3b and 3c. The success probability of the adversary, insofar as the tag has not updated its secrets, is 1.

5. Proposing generalized ultra-lightweight mutual authentication protocols

In this section, we describe the generalized versions of ultra-lightweight mutual authentication protocols (GUMAPs); similar to UMAPSS, we consider only those protocols where reader introduces nonces to the protocol. Depending on which party keeps the history of old data, we consider two cases: a more complex case where both the tag and the reader maintain a record of old data, which is denoted as GUMAP1, and simpler case similar to UMAPSS, where only the reader or the tag does, denoted as GUMAP2. In our description, we denote all the dynamic parameters, which are transferred in a plain-text mode as IDS; all the secret parameters that are updated through the protocol as K; all the static secret parameters by ID; and all the nonces generated by the reader as n. It should be noted that there is no constraint on the length of any of these parameters. Hence, for example, in the case of SLAP [23], where the tag has two keys K^1 and K^2 in our generalized description, we model it as $K = K^1 \parallel K^2$. For a dynamic value $\mathcal{Z}$, its value in the session i is denoted as $\mathcal{Z}^i$, e.g. K^i and IDS^i. Assume that in the beginning of the jth session, the tag and reader record (IDS^{j-1}, K^{j-1}) and (IDS^j, K^j) as old and new records, respectively. The description of GUMAP1, as depicted in Fig. 3, is as follows:

1. The reader R sends Hello to the target tag T.
2. T replies with its IDS^j; if it is not recognized by the reader, R resends Hello to the target tag, and T responds to R with IDS^{j-1}.
3. R uses the received IDS as an index to identify the data related to the tag in the database. If IDS matches the old data of the tag, R uses IDS^{j-1} and K^{j-1} to compute the transferred messages; otherwise, it uses IDS^j and K^j. If IDS does not match any record of the reader's database, the protocol's session is terminated. Assuming that IDS matches a record in the database (we denote it as IDS and K), the reader generates a nonce n^j and computes its challenges $\mathcal{X}^j = f_1(K, n^j, IDS, \ldots)$ and $\mathcal{Y}^j = f_2(K, n^j, IDS, \ldots)$ to be sent to the tag T.
4. T extracts the nonce n^j from the received challenge $\mathcal{X}^j$ and verifies $\mathcal{Y}^j$ using the extracted n^j and its local records of shared parameters with R. If R is authenticated, T responds to R by computing its challenge $Z^j = f_3(K, n^j, IDS, \ldots)$. In addition, the tag updates its parameters as follows:
   - Assigns the current values of K and IDS to K^j and IDS^j, respectively, and updates the tag's shared parameters as follows:

$$IDS^{j+1} = g_1(K, n^j, IDS, \ldots)$$
$$K^{j+1} = g_2(K, n^j, IDS, \ldots)$$

5. R evaluates the challenge Z^j received from T. Assuming that the tag is authenticated, R updates its records for IDS and K to (IDS^j, K^j) and (IDS^{j+1}, K^{j+1}), respectively, in a manner similar to that used by T).

In this protocol, to provide forward security, we assume that the updated value of (IDS^{j+1}, K^{j+1}) is randomized by the introduced value of nonce n^j.

GUMAP2 is a simplified variant of GUMAP1 where only the tag or reader maintains a record of old data, and also where only the reader introduces nonces to the protocol, e.g. UMAPSS, RAPP, and R^{2}AP. In this protocol, without loss of generality, we assume that the last message is sent by the reader to the tag; moreover, the reader should maintain the old records of secret parameters to prevent trivial desynchronization attacks owing to blocking of the last message by the adversary. Hence, in this case, we require an additional round of message transfer over the protocol compared to GUMAP1. The description of GUMAP2, as depicted in Fig. 4, is as follows:

1. The reader R sends Hello to the target tag T.
2. T replies with its IDS^j; if it is not recognized by the reader, R resends Hello to the target tag T, and T responds R with IDS^{j-1}.
3. R uses the received IDS as an index to identify the data related to the tag in the database. If IDS matches the old data of the tag, R uses IDS^{j-1} and K^{j-1} to compute the transferred messages; otherwise, it uses IDS^j and K^j. If IDS does not match any record of the reader's database, the protocol's session is terminated. Assuming IDS matches a record in the database (we denote it as IDS and K), the reader generates a nonce n_1^j and computes challenges $\mathcal{X}^j = f_1(K, n_1^j, IDS, \ldots)$ and $\mathcal{Y}^j = f_2(K, n_1^j, IDS, \ldots)$ to be sent to the tag T.
4. T extracts the nonce n_1^j from the received challenge $\mathcal{X}^j$ and verifies $\mathcal{Y}^j$ using the extracted n_1^j and its local records of shared parameters with R. If R is authenticated, T responds to R by computing the challenge $Z^j = f_3(K, n_1^j, IDS, \ldots)$.
5. R evaluates the received challenge Z^j to authenticate T. Assuming that T is authenticated, R generates another random number n_2^j and responds to T by computing challenges $\mathcal{V}^j = f_4(K, n_1^j, n_2^j, IDS, \ldots)$ and $V_j = f_5(K, n_1^j, n_2^j, IDS, \ldots)$. In addition, the reader updates its parameters by assigning the current values of K and IDS to K^j and IDS^j, respectively, and updates the tag's shared parameters as $IDS^{j+1} = g_1(K, n_1^j, n_2^j, IDS, \ldots)$ and $K^{j+1} = g_2(K, n_1^j, n_2^j, IDS, \ldots)$.

In this protocol, to provide forward security, we assume that the updated value of (IDS^{j+1}, K^{j+1}) is randomized by the introduced values of nonces n_1^j and n_2^j.

5.1. Replay attack on GUMAP1

In this section, we present a replay attack against GUMAP1; this can be applied to RCIA, KMAP, SASI$^+$, and SLAP.

Given that a target tag T records (IDS^{i-1}, K^{i-1}) and (IDS^i, K^i) as the old and new data, respectively, our generalized replay attack operates as follows:

Reader R		**Tag** T
$((IDS^{j-1}, K^{j-1}), (IDS^{j}, K^{j}), ID)$		$((IDS^{j-1}, K^{j-1}), (IDS^{j}, K^{j}), ID)$
	$\xrightarrow{Hello}$	
Generates a nonce n^j	$\xleftarrow{IDS}$	
Calculates $\mathcal{X}^j = f_1(K, n^j, IDS, \ldots)$		
Calculates $\mathcal{Y}^j = f_2(K, n^j, IDS, \ldots)$	$\xrightarrow{\mathcal{X}^j, \mathcal{Y}^j}$	Extracts n^j from $\mathcal{X}^j$
		Verifies $\mathcal{Y}^j$ to authenticate R
		Calculates $\mathcal{Z}^j = f_3(K, n^j, IDS, \ldots)$
	$\xleftarrow{\mathcal{Z}^j}$	Updates IDS and K
Verifies $\mathcal{Z}^j$ to authenticate T		
Updates IDS and K		
$IDS^{j+1} = g_1(K, n^j, IDS, \ldots)$, $K^{j+1} = g_2(K, n^j, IDS, \ldots)$		

Fig. 3. Generalized ultra-lightweight mutual authentication protocol, when both parties maintain a history of old parameters (GUMAP1).

Reader R		**Tag** T
$((IDS^{j-1}, K^{j-1}), (IDS^{j}, K^{j}), ID)$		$((IDS^{j}, K^{j}), ID)$
	$\xrightarrow{Hello}$	
Generates a nonce n^{j_1}	$\xleftarrow{IDS}$	
Calculates $\mathcal{X}^j = f_1(K, n^{j_1}, IDS, \ldots)$	$\xrightarrow{\mathcal{X}^j, \mathcal{Y}^j}$	Extracts n^{j_1} from $\mathcal{X}^j$
Calculates $\mathcal{Y}^j = f_2(K, n^{j_1}, IDS, \ldots)$		Verifies $\mathcal{Y}^j$ to authenticate R
		Calculates $\mathcal{Z}^j = f_3(K, n^{j_1}, IDS, \ldots)$
	$\xleftarrow{\mathcal{Z}^j}$	
Verifies $\mathcal{Z}^j$ to authenticate T		
Generates n^{j_2}		
Calculates $\mathcal{U}^j = f_4(K, n^{j_1}, n^{j_2}, IDS, \ldots)$		
Calculates $\mathcal{V}^j = f_5(K, n^{j_1}, n^{j_2}, IDS, \ldots)$		
Updates IDS and K	$\xrightarrow{\mathcal{U}^j, \mathcal{V}^j}$	Extracts n^{j_2} from $\mathcal{U}^j$
		Verifies $\mathcal{V}^j$ to update IDS and K
$IDS^{j+1} = g_1(K, n^{j_1}, n^{j_2}, IDS, \ldots)$, $K^{j+1} = g_2(K, n^{j_1}, n^{j_2}, IDS, \ldots)$		

Fig. 4. Generalized ultra-lightweight mutual authentication protocol, when only a party maintains a history of old parameter (GUMAP2).

1. **In session** i:
   (a) R receives IDS^i from T, generates n^i, and sends $\mathcal{X}^i = f_1(K^i, n^i, IDS^i, \ldots)$ and $\mathcal{Y}^i = f_2(K^i, n^i, IDS^i, \ldots)$ to T.
   (b) adversary $\mathcal{A}$ eavesdrops IDS^i, $\mathcal{X}^i$, and $\mathcal{Y}^i$; stores them; and terminates the rest of the protocol, to intercept the updating phase of the tag and reader.

2. **In session** $i+1$:

1. $\mathcal{A}$ impersonates R and sends Hello to T, and the tag responds with IDS^i. Note that the tag has not updated its records in the previous session owing to the protocol corruption by $\mathcal{A}$ in Step 1b.
2. When $\mathcal{A}$ receives IDS^i form T, it responds with the eavesdropped $\mathcal{X}^i = f_1(K^i, n^i, IDS^i, \ldots)$ and $\mathcal{Y}^i = f_2(K^i, n^i, IDS^i, \ldots)$ from Step 1b.
3. T authenticates A, sends $Z^i = f_3(K^i, n^i, IDS^i, \ldots)$ to A, and updates its records of shared values to $(IDS^i, K^i) and (IDS^{i+1} = g_1(K^i, n^i, IDS^i, \ldots), K^{i+1} = g_2(K^i, n^i, IDS^i, \ldots))$.

In this attack, the adversary eavesdrops messages in a session, uses them in the subsequent session, and is finally authenticated by the tag. Meanwhile, as long as the tag does not update its records at least twice, it maintains records of shared values (IDS^i, K^i). Hence, the adversary can use the eavesdropped values to apply the replay attack and impersonate the reader. The probability of success of the proposed attack is one.

5.2. Desynchronization attack on GUMAP1

To overcome desynchronization attacks on RAPP [17,19] and its successors, in which only the reader maintains a record of old and new shared parameters, certain protocols – such as RCIA, KMAP, SASI$^+$, and SLAP – recommended that both the reader and tag maintain old parameters. In this section, we present a desynchronization attack against GUMAP1, which can also be applied to the mentioned protocols. Our generalized desynchronization attack operates as follows:

1. **In session** i:
   (a) R receives IDS^i from T, generates n^i, and sends $\mathcal{X}^i = f1(K^i, n^i, IDS^i, \ldots)$ and $\mathcal{Y}^i = f_2(K^i, n^i, IDS^i, \ldots)$ to T.
   (b) T authenticates R, sends $Z^i = f_3(K^i, n^i, IDS^i, \ldots)$ to R, and updates its record of shared values to (IDS^i, K^i) and ($IDS^{i+1} = g_1(K^i, n^i, IDS^i, \ldots)$, $K^{i+1} = g_2(K^i, n^i, IDS^i, \ldots)$).
   (c) R authenticates T and updates its record of shared values to (IDS^i, K^i) and (IDS^{i+1}, K^{i+1}).
   (d) adversary $\mathcal{A}$ eavesdrops IDS^i, $\mathcal{X}^i$, and $\mathcal{Y}^i$ and stores them.

2. **In session** $i+1$:
   (a) R receives IDS^{i+1} from T, generates n^{i+1}, and sends $\mathcal{X}^{i+1} = f_1(K^{i+1}, n^{i+1}, IDS^{i+1}, \ldots)$ and $\mathcal{Y}^{i+1} = f_2(K^{i+1}, n^{i+1}, IDS^{i+1}, \ldots)$ to T.
   (b) $\mathcal{A}$ eavesdrops IDS^{i+1}, $\mathcal{X}^{i+1}$, and $\mathcal{Y}^{i+1}$; stores them; and blocks T to receive $\mathcal{X}^{i+1}$ and $\mathcal{Y}^{i+1}$.
   (c) In this step, the shared parameters of the tag and reader are still (IDS^i, K^i) and (IDS^{i+1}, K^{i+1}), respectively.

3. **In session** $i+2$:
   (a) R sends Hello to T, and the tag replies with IDS^{i+1}.
   (b) $\mathcal{A}$ blocks the tag's response and sends a random value as IDS' to R;
   (c) R does not find a record similar to IDS'. Hence, it sends another Hello to T, and T sends IDS^i to R.
   (d) R receives IDS^i from T, generates n^{i+2}, and sends $\mathcal{X}^i = f_1(K^i, n^{i+2}, IDS^i, \ldots)$ and $\mathcal{Y}^i = f_2(K^i, n^{i+2}, IDS^i, \ldots)$ to T.
   (e) T authenticates R, sends $Z^{i+2} = f_3(K^i, n^{i+2}, IDS^i, \ldots)$ to R, and updates the shared values to (IDS^i, K^i) and ($IDS^{i+2} = g_1(K^i, n^{i+2}, IDS^i, \ldots)$, $K^{i+2} = g_2(K^i, n^{i+2}, IDS^i, \ldots)$).
   (f) R authenticates T and updates the shared values to (IDS^i, K^i) and (IDS^{i+2}, K^{i+2}).

4. **In session** $i+3$:
   (a) $\mathcal{A}$ impersonates R and sends Hello to T, and the tag replies with IDS^{i+1}.
   (b) $\mathcal{A}$ sends another Hello to T, and the tag responds with IDS^i.
   (c) $\mathcal{A}$ receives IDS^i from T and responds with the eavesdropped $\mathcal{X}^i = f_1(K^i, n^i, IDS^i, \ldots)$ and $\mathcal{Y}^i = f_2(K^i, n^i, IDS^i, \ldots)$ from Step 1d.
   (d) T authenticates A, sends $Z^i = f_3(K^i, n^i, IDS^i, \ldots)$ to A, and updates its shared values to (IDS^i, K^i and ($IDS^{i+1} = g_1(K^i, n^i, IDS^i, \ldots)$, $K^{i+1} = g_2(K^i, n^i, IDS^i, \ldots)$).

5. **In session** $i+4$:
   (a) $\mathcal{A}$ once again impersonates R and sends Hello to T, and the tag sends IDS^{i+1}.
   (b) $\mathcal{A}$ receives IDS^{i+1} from T and responds with the eavesdropped $\mathcal{X}^{i+1} = f_1(K^{i+1}, n^{i+1}, IDS^{i+1}, \ldots)$ and $\mathcal{Y}^{i+1} = f_2(K^{i+1}, n^{i+1}, IDS^{i+1}, \ldots)$ from Step 2b.
   (c) T authenticates A, responds with $Z^{i+1} = f_3(K^{i+1}, n^{i+1}, IDS^{i+1}, \ldots)$, and updates its shared values to (IDS^{i+1}, K^{i+1}) and ($IDS^{i+4} = g_1(K^{i+1}, n^{i+1}, IDS^{i+1}, \ldots)$, $K^{i+4} = g_2(K^{i+1}, n^{i+1}, IDS^{i+1}, \ldots)$).

At the end of this attack, the tag's IDS^{old} and K^{old} are equal to $IDS^{i+1} = g_1(K^i, n^i, IDS^i, \ldots)$ and $K^{i+1} = g_2(K^i, n^i, IDS^i, \ldots)$, respectively; moreover, its IDS^{new} and K^{new} are equal to $IDS^{i+4} = g_1(K^{i+1}, n^{i+1}, IDS^{i+1}, \ldots)$ and $K^{i+4} = g_2(K^{i+1}, n^{i+1}, IDS^{i+1}, \ldots)$, respectively. Meanwhile, the reader's IDS^{old} and K^{old} are IDS^i and K^i, respectively, and its IDS^{new} and K^{new} are $IDS^{i+2} = g_1(K^i, n^{i+2}, IDS^i, \ldots)$ and $K^{i+2} = g_2(K^i, n^{i+2}, IDS^i, \ldots)$, respectively. Assuming that the updated values are randomized by the used nonce, it is evident that none of the tag's records matches with any of the reader's records with a high probability. Hence, the reader and tag will be desynchronized after the above attack with a probability of approximately one.

5.3. Replay attack on GUMAP2

Similarly to GUMAP1, with GUMAP2 only the reader contributes to the randomness of the transferred messages. Hence it is convenient to apply a nearly identical attack against GUMAP2. Owing to the similarity, we omit further details. The success probability of this attack is also one. It is evident that none of the tag's records is equal to the reader's records. Hence, the reader and tag will be desynchronized after the above attack with a probability of approximately one.

5.4. Desynchronization attack on GUMAP2

In this section, we explain our generalized desynchronization attack on GUMAP2; it operates as follows:

1. **In session** i:
   (a) R receives IDS^i from T, generates n_1^i, and sends $\mathcal{X}^i = f_1(K^i, n_1^i, IDS^i, \ldots)$ and $\mathcal{Y}^i = f_2(K^i, n_1^i, IDS^i, \ldots)$ to T.
   (b) T authenticates R and sends $Z^i = f_3(K^i, n_1^i, IDS^i, \ldots)$ to R.

(c) R authenticates T, generates another random number n_2^i, sends $\mathcal{U}^i = f_4(K^i, n_1^i, n_2^i, IDS^i, \ldots)$ and $\mathcal{V}^i = f_5(K^i, n_1^i, n_2^i, IDS^i, \ldots)$ to T, and updates its record of shared values to (IDS^i, K^i) and $(IDS^{i+1} = g_1(K^i, n_1^i, n_2^i, IDS^i, \ldots), K^{i+1} = g_2(K^i, n_1^i, n_2^i, IDS^i, \ldots))$.

(d) Adversary $\mathcal{A}$ eavesdrops IDS^i, $\mathcal{X}^i$, $\mathcal{Y}^i$, $\mathcal{U}^i$, and $\mathcal{V}^i$ and stores them. Moreover, $\mathcal{A}$ blocks $\mathcal{U}^i$ and $\mathcal{V}^i$. Hence, the tag's record remains (IDS^i, K^i).

2. **In session** $i+1$:

(a) R receives IDS^i from T, generates n_1^{i+1}, and sends $\mathcal{X}^{i+1} = f_1(K^i, n_1^{i+1}, IDS^i, \ldots)$ and $\mathcal{Y}^{i+1} = f_2(K^i, n_1^{i+1}, IDS^i, \ldots)$ to T.

(b) T authenticates R and sends $Z^{i+1} = f_3(K^i, n_1^{i+1}, IDS^i, \ldots)$ to R.

(c) R authenticates T, generates another random number n_2^{i+1}, sends $\mathcal{U}^{i+1} = f_4(K^i, n_1^{i+1}, n_2^{i+1}, IDS^i, \ldots)$ and $\mathcal{V}^{i+1} = f_5(K^i, n_1^{i+1}, n_2^{i+1}, IDS^i, \ldots)$ to T, and updates its shared values to (IDS^i, K^i) and $(ID_{i+2} = g_1(K^i, n_1^{i+1}, n_2^{i+1}, IDS^i, \ldots), K^{i+1} = g_2(K^i, n_1^{i+1}, n_2^{i+1}, IDS^i, \ldots))$.

(d) $\mathcal{A}$ blocks $\mathcal{U}^{i+1}$ and $\mathcal{V}^{i+1}$. Hence, the tag's record remains (IDS^i, K^i).

3. **In session** $i+2$:

1. $\mathcal{A}$ impersonates R and sends Hello to T, and the tag sends IDS^i.
2. $\mathcal{A}$ sends the eavesdropped $\mathcal{X}^i = f_1(K^i, n_1^i, IDS^i, \ldots)$ and $\mathcal{Y}^i = f_2(K^i, n_1^i, IDS^i, \ldots)$ from Step 1d to T.
3. T authenticates $\mathcal{A}$ as the legitimate R and sends $Z^i = f_3(K^i, n_1^i, IDS^i, \ldots)$ to the expected R.
4. $\mathcal{A}$ sends the eavesdropped $\mathcal{U}^i = f_4(K^i, n_1^i, n_2^i, IDS^i, \ldots)$ and $\mathcal{V}^i = f_5(K^i, n_1^i, n_2^i, IDS^i, \ldots)$ to T.

At the end of this attack, the tag's IDS and K are $IDS^{i+1} = g_1(K^i, n_1^i, n_2^i, IDS^i, \ldots)$ and $K^{i+1} = g_2(Ki, n_1^i, n_2^i, IDS^i, \ldots)$, respectively. However, the reader's IDS^{old} and K^{old} are IDS^i and K^i, respectively, and its records for IDS^{new} and K^{new} are $IDS^{i+2} = g_1(K^i, n_1^{i+1}, n_2^{i+1}, IDS^i, \ldots)$ and $K^{i+2} = g_2(K^i, n_1^{i+1}, n_2^{i+1}, IDS^i, \ldots)$, respectively.

It is evident that none of the tag's records is equal to the reader's records. Hence, the reader and tag will be desynchronized after the above attack with a probability of approximately one.

6. Description of MAC algorithm $f_{IV_0}^{DM-\chi per}(.)$

In order to solve security issues in the cited ultra-lightweight protocols and present a more secure module that can be able to meet the requirement of lightweight protocols, we propose a new efficient permutation-based message authentication code (MAC) function and utilize this function in our protocol. In this section, we explain the details of our MAC denoted by $f_{IV_0}^{DM-\chi per}(.)$.

Our proposed MAC $f_{IV_0}^{DM-\chi per} : \{0,1\}^* \rightarrow \{0,1\}^{3n}$ for a fixed initial value IV_0 takes an arbitrary-length message (denoted by M) and produces a $3n$-bit tag value. The input message is padded to obtain a multiple of $3n$ bits and then processed in b blocks $m_i, i = 1, \ldots, b$ of $3n$ bits based on the well-known Merkle–Damgård iterative structure [27]. A schematic view of $f_{IV_0}^{DM-\chi per}(.)$ for message $M = m_1 \| \cdots \| m_b$ is depicted in Fig. 5. More precisely, as shown in Fig. 5, each iteration invokes a one-way Davies Meyer (DM)-based compression function (denoted by $DM - \chi per^{\mathcal{R}}(.)$) which takes a $3n$-bit initial value (IV_{i-1}) and also a $3n$-bit message block m_i and outputs another $3n$-bit chaining value (denoted IV_i). In our proposed MAC, the initial value IV_0 is a set of fixed constants, and the final chaining value is the tag of the $b \times 3n$-bit message blocks $M = (m_1 \| m_2 \| \cdots \| m_b)$ (see Fig. 5). In the following we present the details of function $\chi per^{\mathcal{R}}(.)$ used in $DM - \chi per^{\mathcal{R}}(.)$ compression function.

6.1. Description of $\chi per^{\mathcal{R}}(.)$

As it can be seen in Fig. 5, each iteration invokes a DM-based compression function which it uses $\chi per^{\mathcal{R}}(.)$ function in its structure. $\chi per^{\mathcal{R}}(.)$ function is based on a SPN structure and accepts a $3n$-bit data block m_i and a $3n$-bit initial value IV_{i-1} as inputs, and outputs a $3n$-bit. An overview of $\chi per^{\mathcal{R}}(.)$ is illustrated in Fig. 6. As one can be seen from Fig. 6, $\chi per^{\mathcal{R}}(.)$ uses an expansion algorithm (EA) to convert $3n$-bit message m_i to $3 \times (\mathcal{R}+1)$ bits updated message. The details of the expansion algorithm can be seen in Fig. 7. According to this algorithm, the $3n$-bit data block m_i is split into three n-bit messages, and then the sub-messages for each round are derived based on a LFSR structure (see Fig. 7). $\chi per^{\mathcal{R}}(.)$ also iterates a $\mathcal{R}$-round permutation-based structure denoted by $\chi per(.)$ to compute its output (see Fig. 6). In the following, we explain the structure of $\chi per(.)$.

6.2. Description of $\chi per(.)$ [28]

A schematic view of $\chi per(.)$ permutation is depicted in Fig. 8. As one can see from Fig. 8, each round function $\chi per(.)$ applies the following two operations: non-linear, and linear layers. Note that the non-linear layer of $\chi per(.)$ uses a nonlinear function χ which is used in the Keccak [29]. χ function is an adjustable permutation for any odd value and we use a variant with 3 bits input–output.

6.3. Our choice for n and $\mathcal{R}$

The variables n and $\mathcal{R}$ provides trade off between efficiency and security. Our recommendations for n is 32 bits, and $\mathcal{R} = 16$ for the first and last blocks (i.e., m_1 and m_b), and $\mathcal{R} = 8$ for other middle blocks (see Fig. 5).

In Appendix A, a schematic view to generate a $3n$-bit tag for message $M = (m_1 \| m_2)$ under initial value IV_0 is illustrated. Also, more details of message authentication code $f_{IV_0}^{DM-\chi per}(M)$ as algorithm-base is presented in Fig. B.12 in Appendix B.

6.4. Security analysis of our proposed MAC

In Appendix C, the security of $\chi per^{\mathcal{R}}(.)$ function has been investigated against several well-known attacks e.g., differential [30], related-key differential [31], linear [32], impossible differential [33,34], and zero-correlation [35] cryptanalysis. In addition, we explain a performance analysis of the proposed MAC further in Section 9.

Our security and performance analysis of the proposed MAC shows that it can be use in any lightweight authentication protocol. Given the message authentication code $f_{IV_0}^{DM-\chi per}(.)$, we present an improved protocol GIMAP in the next section. It is worth emphasizing that we set the initial value IV_0 with the secret key K in our proposed protocol.

7. Improved protocol - GIMAP

This section contains a description of how we improve GUMAP in order to provide an acceptable security level against the attacks. We follow the same notation and call the improved protocol as GIMAP, i.e., generalized improved mutual authentication protocol. The main modification is to compel both the protocol parties to contribute to the randomness of the transferred messages. In addition, our analysis on $f_{IV_0}^{DM-\chi per}(.) : \{0,1\}^{b \times 3n} \rightarrow \{0,1\}^{3n}$ in Appendix C shows that it is a secure random function. Therefore, we should have

$$\forall (x_1 \neq x_2) \ \Pr\left(f_{IV_0}^{DM-\chi per}(x_1) = f_{IV_0}^{DM-\chi per}(x_2)\right) = 2^{-3n}.$$

This property will avoid the most possible attacks that we discuss further in Section 8.

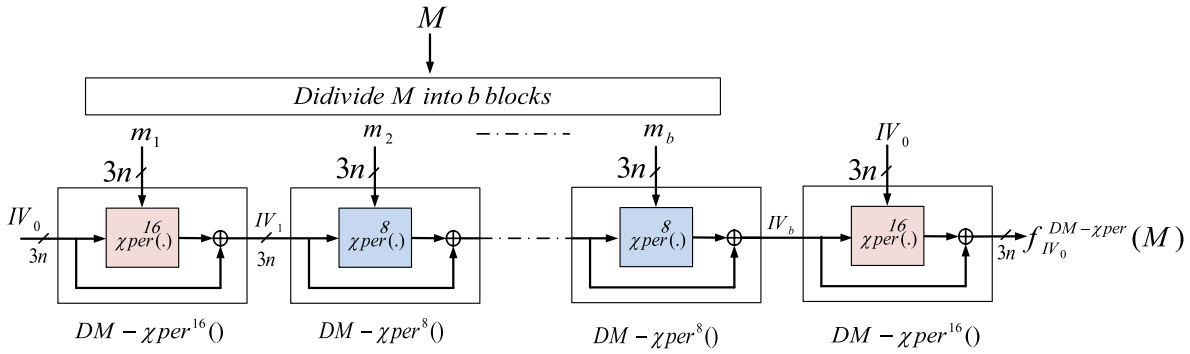

Fig. 5. An overview of $f_{IV_0}^{DM-\chi per}(M)$ structure.

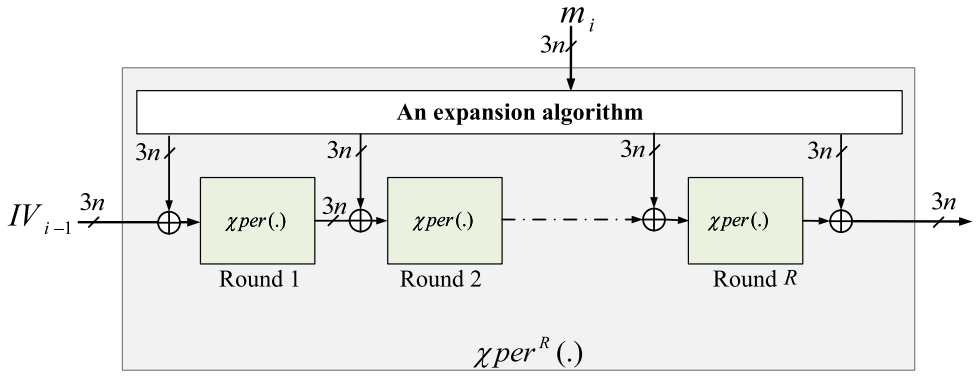

Fig. 6. An overview of $\chi per^R(.)$ structure.

Expansion algorithm: $EA : \{0,1\}^{3n} \rightarrow \{0,1\}^{3\times(\mathcal{R}+1)}$

$EA : m = (m^0||m^1||m^2) \rightarrow \{m_{r,i}\}, r = 1, \cdots, \mathcal{R}+1; i = 0,1,2.$

1. Set $w_0 = m^0, w_1 = m^1, w_2 = m^2$.
2. For $i = 0$ to $3\mathcal{R} - 1$

   $w_{i+3} = (w_i \lll 4) \oplus (w_{i+1} \oplus c_i)^{\dagger}$.
3. For $r = 1$ to $\mathcal{R} + 1$

   $m_{r,0} = w_{3i-3}, m_{r,1} = w_{3i-2}, m_{r,2} = w_{3i-1}$.

$\dagger$: The $(3n) \times \mathcal{R}$-bit round constants are all first $((3n) \times \mathcal{R})/4$ hexadecimal digits of π $(i.e., \texttt{243F6A8...})$

Fig. 7. The details of expansion algorithm using in $\chi per^R(.)$ structure.

7.1. Description of GIMAP protocol

In GIMAP, both the reader and tag introduce nonces to the protocol, and only the tag maintains a record of old data.

Assume that at the beginning of the jth session of the protocol, the tag's old and new records of IDS and K are (IDS^{j-1}, K^{j-1}) and (IDS^j, K^j), and that the reader maintains only the updated value, which is (IDS^j, K^j). The description of GIMAP as depicted in Fig. 9 is as follows:

1. The reader R sends Hello to the target tag T.
2. T generates a random number n_t^j and sends it with IDS^j to the reader; if IDS is not recognized by the reader, R resends Hello to T, and T responds with IDS^{j-1} and a new fresh n_t^j.
3. R uses the received IDS as an index to identify the data related to the tag in the database. If IDS does not match any record of the reader's database, the session is terminated. Assuming IDS matches a record in the database (we denote it as IDS and K), the reader generates a nonce n_r^j, computes the challenge $\mathcal{X}^j = f_K^{DM-\chi per}(n_t^j, n_r^j, IDS)$, and sends them to T.
4. Given n_t^j, n_r^j, and the local parameters, T verifies the received $\mathcal{X}^j$ to authenticate R. If R is authenticated, T responds to R by computing its challenge $\mathcal{Y}^j = f_K^{DM-\chi per}(, n_r^j, n_t^j, IDS)$. In addition, the tag assigns the current values of K and IDS to K^j

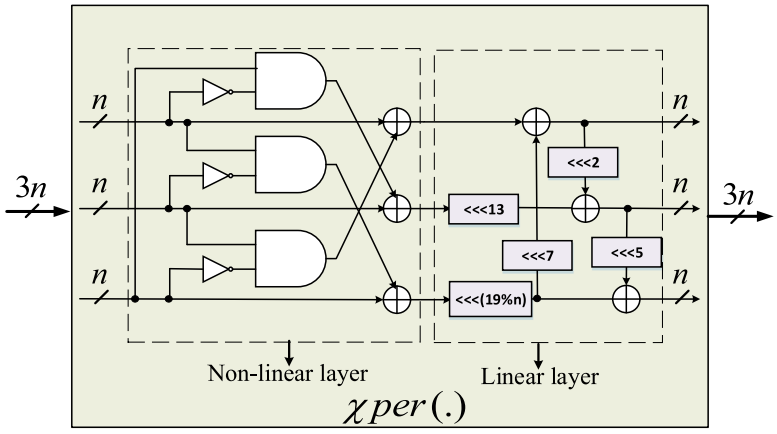

Fig. 8. $\chi per(.)$ permutation [28].

Reader R		**Tag** T
$((IDS^j, K^j), ID)$		$((IDS^{j-1}, K^{j-1}), (IDS^j, K^j), ID)$
	$\xrightarrow{Hello}$	
Generates a nonce n_r^j	$\xleftarrow{IDS, n_t^j}$	Generates a nonce n_t^j
Calculates $\mathcal{X}^j = f_K^{DM-\chi per}(n_t^j, n_r^j, IDS)$	$\xrightarrow{\mathcal{X}^j, n_r^j}$	Verifies $\mathcal{X}^j$ to authenticate R
		Calculates $\mathcal{Y}^j = f_K^{DM-\chi per}(n_r^j, n_t^j, IDS)$
Verifies $\mathcal{Y}^j$ to authenticate T	$\xleftarrow{\mathcal{Y}^j}$	Updates IDS and K
Updates IDS and K		
$IDS^{j+1} = f_K^{DM-\chi per}(IDS, n_t^j, n_r^j), K^{j+1} = f_K^{DM-\chi per}(n_t^j, n_r^j).$		

Fig. 9. Generalized Improved mutual authentication protocol (GIMAP).

and IDS^j, respectively, and updates the tag's shared parameters as follows:

$$IDS^{j+1} = f_K^{DM-\chi per}(IDS, n_t^j, n_r^j)$$
$$K^{j+1} = f_K^{DM-\chi per}(n_t^j, n_r^j)$$

5. R evaluates the challenge received $\mathcal{Y}^j$ from T. Assuming that the tag is authenticated, R updates its records for IDS and K to (IDS^{j+1}, K^{j+1}) in a manner similar to that adopted by T.

It should be noted that the proposed modification by itself could not improve the security level of the protocols mentioned in this paper. For example, in [19], a secret disclosure attack against RAPP was presented using the shortcoming of its permutation function. Therefore, to have a secure protocol, it is necessary to use secure components that may increase the cost of the implementation. However, novel achievements in the field of lightweight cryptography provide us with numerous options of lightweight algorithms to match the application constraints; $f_{IV_0}^{DM-\chi per}(.)$ is an example of similar lightweight primitives.

8. Evaluating the security of GIMAP

In this section, we assess the robustness and resistance of GIMAP against various kinds of attacks. In order to evaluate the security level of the proposed protocol, both informal and formal methods are utilized. Therefore, we prove that GIMAP can solve the security issues of previous protocols as well as provide an acceptable security level.

8.1. Informal security analysis of GIMAP

In this section, we demonstrate the security of the improved protocol against various attacks as follows:

8.1.1. Secret disclosure attack

The only secret value that is shared between the tag and reader is K. However, it is not feasible for the adversary to reach K without compromising a tag (which is out of the scope of this design) or invert the random/MAC functions $DM - \chi per^R(.)$ in the structure of $f_K^{DM-\chi per}(.)$. Our analysis in Appendix C shows DM-χper adequately secure, e.g. a secure MAC function, it is not feasible to invert it. Hence, the proposed protocol is secure against secret disclosure attacks.

8.1.2. Replay attack

The messages transferred in the jth session are $(Hello; IDS; \mathcal{X}^j; \mathcal{Y}^j)$; here, $\mathcal{X}^j = f_K^{DM-\chi per}(n_t^j, n_r^j, IDS)$ and $Yj = f_K^{DM-\chi per}(n_r^j, n_t^j, IDS)$. Given that n_r^j and n_t^j are generated randomly and are changed in each session, the adversary cannot regulate both n_t^j and n_r^j simultaneously. Therefore, since the protocol utilizes fresh value of n_t^j and n_r^j in a new session, even if the adversary eavesdrops the messages and uses in the subsequent session, the protocol will abort the connection. Hence, the proposed protocol is secure against replay attacks.

8.1.3. Impersonation attack

Given that the proposed protocol is secure against replay attacks, the only method to impersonate the tag or reader is to produce a valid $\mathcal{X}^j/\mathcal{Y}^j$. Because the function $f_K^{DM-\chi per}$ is secure, for generating a valid $\mathcal{X}^j/\mathcal{Y}^j$, the adversary must forge a secure MAC which does not appear doable.

Table 2
BAN logic notations and rules that are used in this paper.

Notations	Description
$\sharp(X)$	X is fresh
$\{X\}_K$	The encryption of X using K as the key
$P \lhd X$	P receives the message X
$P \overset{K}{\longleftrightarrow} Q$	K is the shared secret between P and Q
$R1: \frac{P\vert \equiv P \overset{K}{\longleftrightarrow} Q, P \lhd \{X\}_K}{P\vert \equiv Q\vert \sim X}$	If P believes K is securely shared between P and Q and also sees the $\{X\}_K$ the P believes Q conveys the message X
$R2: \frac{P\vert \equiv Q\vert \sim \{X, Y, Z, \ldots\}}{P\vert \equiv Q\vert \sim \{X\}}$	If P believes that Q has sent a series of messages, He also believes that Q had sent each individual message also.

Table 3
BAN logic expression of GIMAP messages.

Messages No.	Description	Messages No.	Description
M_1	$T \lhd Hello$	M_3	$T \lhd \mathcal{X}^j = f_K^{DM-\chi per}(n_t^j, n_r^j, IDS), n_r^j$
M_2	$R \lhd IDS, n_t^j$	M_4	$R \lhd \mathcal{Y}^j = f_K^{DM-\chi per}(n_r^j, n_t^j, IDS)$

Table 4
GIMAP messages idealization.

Idealized messages No.	Description
IM_3	$T \lhd \mathcal{X}^j = \{n_t^j, n_r^j, IDS\}_K$
IM_4	$R \lhd \mathcal{Y}^j = \{n_r^j, n_t^j, IDS\}_K$

Table 5
GIMAP assumption and security goals.

Assumption/Goal No.	Description
$A1$	$T\vert \equiv \sharp n_t^j$
$A2$	$R\vert \equiv \sharp n_r^j$
$A3$	$T\vert \equiv T \overset{IDS^j, K^j, ID}{\longleftrightarrow} R$
$A4$	$R\vert \equiv R \overset{IDS^j, K^j, ID}{\longleftrightarrow} T$
$G1$	$T\vert \equiv R\vert \sim \sharp n_r^j$
$G2$	$R\vert \equiv T\vert \sim \sharp n_t^j$

Table 6
GIMAP security goals deduction.

Deduction No.	Given messages and assumptions	Used rule	Deduction Description	Goal No.
$D1$	$IM3, A1$	$R1$	$T\vert \equiv R\vert \sim \{n_t^j, n_r^j, IDS\}$	–
$D2$	$D1$	$R2$	$T\vert \equiv R\vert \sim \{n_r^j\}$	$G1$
$D3$	$IM4, A4$	$R1$	$R\vert \equiv T\vert \sim \{n_r^j, n_t^j, IDS\}$	–
$D4$	$D3$	$R2$	$R\vert \equiv T\vert \sim \{n_t^j\}$	$G2$

8.1.4. Desynchronization attack

In the improved protocol, in order to desynchronize the tag and reader the adversary should be capable of either impersonating the tag or reader or blocking the last message. Because the tag will update its records and maintain the history of old data in this case, the reader and tag can be re-synchronized based on those records. Hence, the proposed protocol is secure against desynchronization attacks.

8.2. Formal security analysis of GIMAP

Many formal methods through mathematical or logical relations have been developed to evaluate the robustness of a cryptographic protocol. Such formal methods are either manual, as with GNY logic [36] and BAN logic [37] or automatic, as with Proverif [38] and Scyther [39]. We evaluate here the security of GIMAP using a manual method and an automatic tool, which are BAN logic and the Scyther tool respectively (widely accepted in the literature) to verify the security of a cryptographic protocol.

8.2.1. Formal security analysis of GIMAP through BAN logic

The security evaluation of any protocol (including GIMAP) with BAN logic includes several steps. In the first step, the messages of the protocol are depicted in the BAN logic form. Next, the protocol messages are represented in an idealized form, where the messages that are plain or do not increase the confidentiality are deleted. Afterward, the security assumptions and security goals of the protocol are expressed. Finally, based on BAN logic rules, the security goals may be deduced from idealized messages and also the protocol assumptions. Table 2 represents the notations used in BAN logic to prove the security of a protocol. Here, by using BAN logic the robustness of GIMAP as a secure authentication protocol will be improved as follows:

- Protocol messages are written using mathematical relationships and logic symbols. Table 3 represents the result of this step for GIMAP.
- Simple and unencrypted messages are deleted. Table 4 shows the result of this step for GIMAP.
- The assumptions in the protocol, as well as the security objectives that we are seeking to substantiate, are expressed. Table 5 shows the result of this step for GIMAP.
- Finally, using messages and protocol assumptions and based on the existing rules of BAN logic, we try to deduce the desired security objectives. Table 6 shows the results of this step for GIMAP.

As presented in Table 6, if we consider $IM3$ and $A3$ then, based on $R1$ rule of BAN logic, we can deduce that $D1: T\vert \equiv R\vert \sim \{n_t^j, n_r^j, IDS\}$. Using $D1$ and based on $R2$, we deduce that $D2: T\vert \equiv R\vert \sim \{n_r^j\}$, which satisfies $G1$ security goal. $G1$ indicates that what T receives is the same as the message that R has sent. Moreover, the message has not been forged and has not changed during the transfer.

If we consider $IM4$ and $A4$ then, based on $R1$ rule of BAN logic, we deduce that $D3: R\vert \equiv T\vert \sim \{n_r^j, n_t^j, IDS\}$. Using $D3$ and based on $R2$, we can deduce that $D4: R\vert \equiv T\vert \sim \{n_r^t\}$, which satisfies $G2$ security goal. $G2$ indicates that R receives the same message sent by T. Moreover, this message has not been forged and has not changed during the transfer. Hence, based on BAN logic, GIMAP provides the desired security goals.

8.2.2. Formal security evaluation by the Scyther tool

We used the Scyther tool to verify that GIMAP satisfies the purported security for the proposed protocol. The Scyther tool automatically examines all defined security claims in the proposed protocol, and should any kind of attack be found, the scenario is graphically depicted. More precisely, Scyther investigates secrecy and authentication in the security protocols. To this end, the selected protocol is described in the SPDL language and then analyzed by the Scyther tool. The results of the GIMAP verification are represented in Fig. 10. As we can see, the Scyther tool has not detected any vulnerability in this protocol (i.e. no attacks within bounds). Hence, we can argue that GIMAP provides the desired security level.

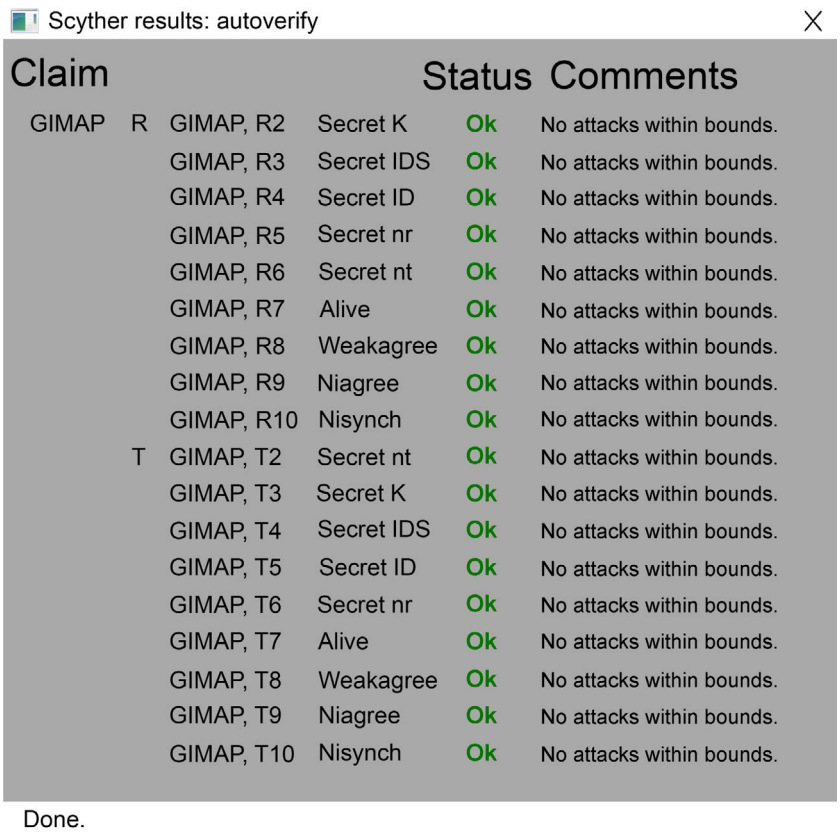

Fig. 10. The Scyther tool security verification results of GIMAP.

Table 7
Throughput and implementation cost for various functions [28,41,42]. *tp*: throughput; *tp/area*: throughput-area ratio; LUTs: look-up-tables; Mhz: Mega Hertz; Mbps: Mega bit per second.

Function	*area* (LUT)	*frequency* (Mhz)	*tp* (Mbps)	*tp/area* (Mbps/LUT)
SIMON-96	435	564	1041	2.39
SPECK-96	452	473	1622	3.59
$\chi per^{\mathcal{R}}(\cdot)$-96	460	680	4080	8.86

9. Performance and security comparison

9.1. Performance analysis of $\chi per^{\mathcal{R}}(\cdot)$

The MAC $f_K^{DM-\chi per}(.)$ is the main function to generate a tag value in the structure of GUMAP scheme. To evaluate the efficiency of the proposed MAC, the $\chi per^{\mathcal{R}}(\cdot)$ function only need to be implemented. As before mentioned, security and performance of the $\chi per^{\mathcal{R}}(\cdot)$ function depends on the two parameters n and $\mathcal{R}$ that in our paper are considered as $n = 32$ and $\mathcal{R} = 8$, and 16. In [28], an implementation of $\chi per^{\mathcal{R}}(\cdot)$ function on the FPGA module Xilinx Kintex-7 using the hardware description language VHDL is performed. The throughput *tp*, and the throughput-area ratio *tp/area* of the $\chi per^{\mathcal{R}}(\cdot)$ algorithm for $n = 32$, and $\mathcal{R} = 16$ are computed by the following formula [28]:

$$tp = \frac{Block\ size}{Cycles\ per\ block} \times Frequency(Mhz)$$

$$tp/area = \frac{tp}{Slice\ LUTs}$$

Table 7 shows the throughput and implementation cost comparison of the $\chi per^{\mathcal{R}}(\cdot)$ function with some known lightweight encryption functions which are used in the lightweight authentication schemes [40].

According to Table 7, the device utilization of the simulation after synthesis of the $\chi per^{\mathcal{R}}(\cdot)$ is 460 look-up-tables (LUTs), Cycles per block is 16 and its clock rate ($frequency$) is 680(Mhz). In addition, $\chi per^{\mathcal{R}}(\cdot)$ function has the highest $tp/area$ which shows that it is more lightweight than the others. Also, a logic diagram of the synthesized $\chi per^{\mathcal{R}}(\cdot)$ function in terms of logic gates such as AND, NAND, and OR is presented in [28].

9.2. Performance analysis of GIMAP and security comparison

In order to highlight the differences in performance and security between GIMAP and other UMAP protocols, we propose a synthesis in Table 8. For the performance assessment of the protocols under the same conditions, we assume that L stands for 96-bits (which corresponds to the length of variables in EPC Tag Data Standard) [43]. In GIMAP, each tag stores (IDS^{j-1}, K^{j-1}), (IDS^j, K^j) and ID, and certainly needs 5L, that is equal to the minimum amount of memory requirements compared to other protocols. In addition, when looking at the communication overhead in analyzing the number of messages required in a mutual authentication process GIMAP shows an acceptable situation, as only 4L are transferred. In terms of security, GIMAP is resistant against various kinds of IoT attacks (traceability, desynchronization, replay, and impersonation). The information summarized in Table 8 allows us to position GIMAP relative to the other protocols mentioned and shows that GIMAP can better satisfy the security requirement for a lightweight mutual authentication protocol.

10. Conclusion

By their complex nature (multi-features, multi-uses), IoT devices combine physical processes with cyber connectivity, and raises a number of challenges related to how safe connected things are (or should

Table 8
The comparison of GIMAP with other UMAP protocols. # MoT: Memory on tag; #Mfma: Messages for mutual authentication; Rtta: Resistance to traceability attacks; Rtda: Resistance to desynchronization attacks; Rtra: Resistance to replay attacks; Rti: Resistance to impersonation attacks.

Protocols	RAPP	R2AP	RCIA	KMAP	SLAP	SASI+	UMAPSS	GIMAP
Category	GUMAP2	GUMAP2	GUMAP1	GUMAP1	GUMAP1	GUMAP1	GUMAP2	–
#MoT	5L	5L	7L	7L	7L	7L	11L	5L
#Mfma	5L	7L	5L	4L	4L	4L	4L	4L
Rtta	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Rtda	No	No	No	No	No	No	No	Yes
Rtra	No	N	No	No	No	No	No	Yes
Rti	No	Yes	No	No	No	No	Yes	Yes

be); imposing to discover and address their vulnerabilities in term of security, confidentiality, identity, and privacy [5].

In this study, we have evaluated the security level of some recent ultra-lightweight mutual authentication protocols, which we grouped under a generalized version called GUMAPs. Thereafter, we classified them into two categories: (i) GUMAP1, where both parties maintain a history of old parameters; and (ii) GUMAP2, where only one party maintains a history of old parameters. We then established that both the groups are vulnerable to replay and desynchronization attacks, and proposed a more secure generalized improved mutual authentication protocol (GIMAP) against the above-mentioned attacks. Our analysis showed that, regardless of the information within messages, any protocol with a similar framework is vulnerable to a desynchronization attack. In addition, we demonstrated that all protocol parties should contribute to the session randomization; otherwise, the underlying protocol would be vulnerable to a desynchronization attack. Finally, to overcome attacks and assuming that there is a secure cryptographic preventive, we proposed the sketch of a protocol (GIMAP), which could prove secure against the proposed attacks.

The security and performance comparison results of the GIMAP show that this protocol is well suited for resource-constrained environments such as RFID tags and IoT devices. In addition, to propose GIMAP protocol, we designed a new lightweight MAC which can be used in any other protocol independent of GIMAP. In this paper, we have shown its security against various attack, but we encourage other researchers to investigate its security independently.

Finally, but certainly not least, we welcome researchers to use GIMAP and other related protocols in a real-world application to compare the hidden properties of those protocols. In our paper, in order to offer a fair comparison and to be consistent with other research, we only have compared the protocols primitives as a rule of thumb. The implementation of a protocol, on the other hand, involves many application-specific features that are outside the scope of this work. As a further research, for challenging its performance, a team could adapt the GIMAP protocol to a real-world application and compare it to other relevant protocols.

Author statement

It should be noted an early draft of the contribution 2 (i.e., a partial and shorter version of this article limited to the discussion of desynchronization attack) has been hosted on Cryptology ePrint Archive as a technical report [26].

CRediT authorship contribution statement

Masoumeh Safkhani: Conception and design of study, Writing – original draft. **Samad Rostampour:** Conception and design of study, Writing – original draft. **Ygal Bendavid:** Conception and design of study, Writing – original draft. **Sadegh Sadeghi:** Conception and design of study, Writing – original draft. **Nasour Bagheri:** Conception and design of study, Writing – original draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

Approval of the version of the manuscript to be published (the names of all authors must be listed): Masoumeh Safkhani Samad Rostampour Ygal Bendavid, Sadegh Sadeghi Nasour Bagheri.

Appendix A. Schematic view of MAC $f_{IV_0}^{DM-\chi per}(m_1, m_2)$

A schematic view of the $f_{IV_0}^{DM-\chi per}(m_1, m_2)$ to generate $3n$ bits tag is illustrated in Fig. A.11.

Appendix B. Code-based algorithm of MAC $f_{IV_0}^{DM-\chi per}$

A code-based algorithm of MAC $f_{IV_0}^{DM-\chi per}(M)$, $(|M| = b \times 3n)$ to generate a $3n$-bit tag can be seen in Fig. B.12.

Appendix C. Security analysis of χper function

To analysis of our proposed Mac, we should analysis the function $\chi per^{\mathcal{R}}(\cdot)$. This can be observed from Fig. 5. Therefore, for more detailed analysis we can consider the $\chi per^{\mathcal{R}}(\cdot)$ function as a block cipher that takes $3n$-bit plaintext, $3n$-bit master key, and runs for $\mathcal{R}$ rounds to produce a $3n$-bit ciphertext from the plaintext (see Fig. 6). Hence, the expansion algorithm (EA) of $\chi per^{\mathcal{R}}(\cdot)$ (Fig. 7) can be consider as a key schedule algorithm. In this section, we present the results of our security analysis of $\chi per^{\mathcal{R}}(\cdot)$ against differential [30], related-key differential [31], linear [32], impossible differential [33,34], and zero-correlation [35] cryptanalysis. To investigate these attacks, the action of the Non-linear layer is described as parallel with a 3×3 S-box. This S-box in hexadecimal notation is given by Table C.9.

C.1. Differential/linear cryptanalysis

In order to argue for the resistance of $\chi per^{\mathcal{R}}(\cdot)$ against differential (Related Key (RK) differential and linear attacks, we applied Mixed Integer Linear Programming (MILP) method as explained in [44–46] to search for differential and linear characteristics. The results based on the number of active S-boxes are listed in Table C.10.

The maximum differential/linear probability of the 3-bit S-box used in $\chi per^{\mathcal{R}}(\cdot)$ cipher is 2^{-2}. In differential attack, if the number of active S-boxes in the initialization rounds is at least 49 $(> 3n/2)$, we consider the cipher to be resistant against differential cryptanalysis. In linear attack, assuming the number of active S-boxes is s, according to Piling-Up Lemma [32], the bias ϵ is $2^{s-1} \times (2^{-2})^s$. Therefore, to make the algorithm resistant against linear attack it needs: $\frac{1}{\epsilon^2} = 2^{2(s+1)} > 2^{3n}$. Thus, for $n = 32$, the number of linearity active S-boxes must be at least 49.

Hence, the result of Table C.10 shows differential and linear cryptanalysis are used to attack a few more rounds than the number of rounds of the characteristic. Also, Table C.10 shows our results hold in the related-key setting.

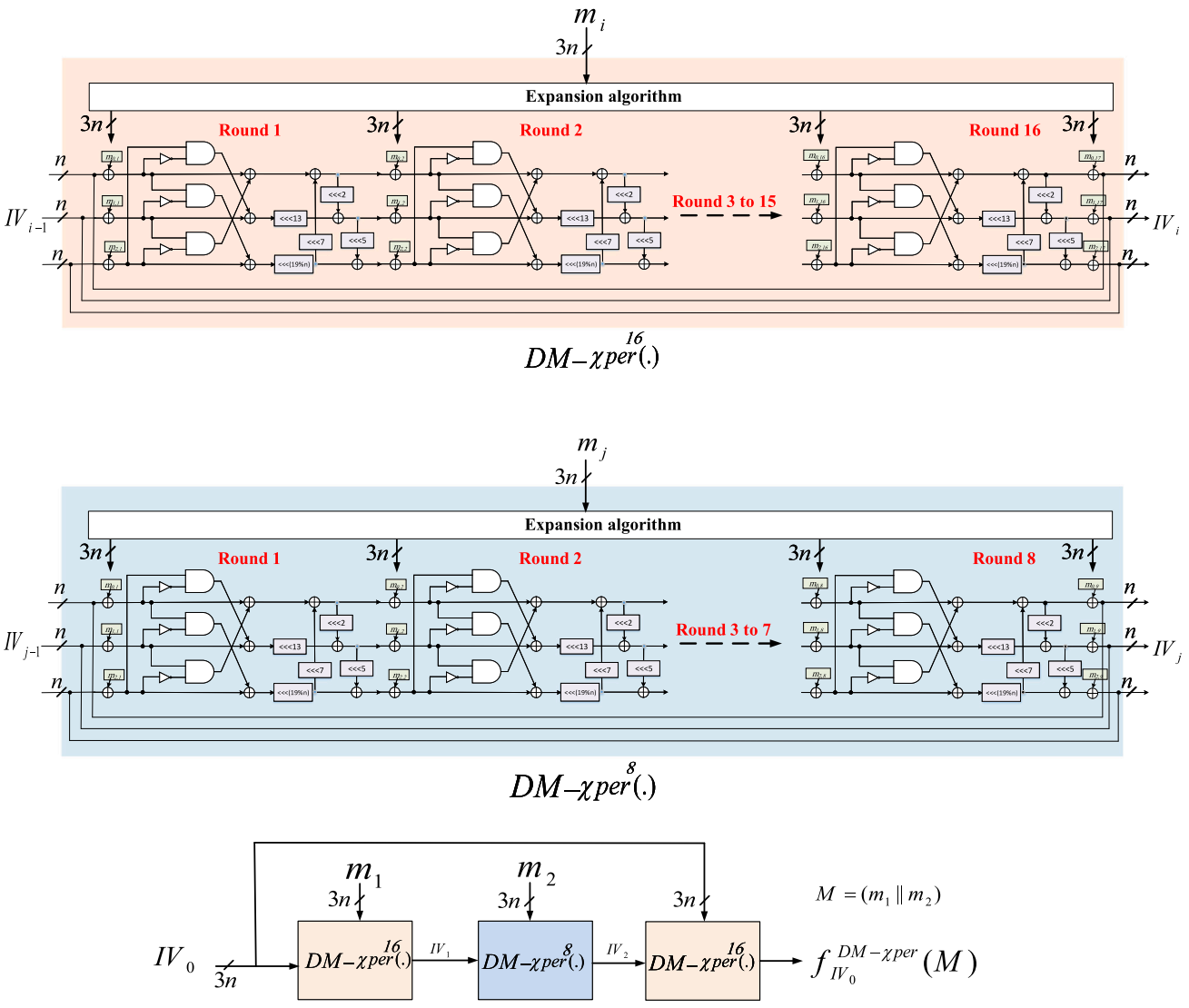

Fig. A.11. A schematic view of the message authentication code $f_{IV_0}^{DM-\chi per}(M)$, ($|IV_0| = 3n, |M| = 6n$) with the details of its compression function $DM - \chi per^R$. The details of the expansion algorithm is explained in Fig. 7.

Table C.9
The 3-bit S-box used in $\chi per^{\mathcal{R}}(\cdot)$ in hexadecimal form.

x	0	1	2	3	4	5	6	7
$S(x)$	0	3	6	1	5	4	2	7

Table C.10
Lower bounds on the number of active S-boxes in $\chi per^{\mathcal{R}}(\cdot)$. In case the MILP optimization was too long, we provide upper bounds between parentheses.

♯ rounds ($\mathcal{R}$)	1	2	3	4	5	6	7	8	9	10	11	12
Linear	1	3	6	11	19	24	28	32	(36)	(40)	(44)	(48)
Differential	1	3	6	11	18	24	(31)	(37)	(45)	(50)	–	–
RK-Differential	0	1	3	7	18	(23)	(33)	(39)	(47)	–	–	–

C.2. Impossible differential characteristics

Impossible differential attack [33,34] finds two internal state differences Δ_i, Δ_o such that Δ_i is never propagated to Δ_o. The attacker then finds many pairs of plaintext/ciphertext and key values leading to (Δ_i, Δ_o). Those key values are wrong values, thus key space can be reduced. To search for impossible characteristics we applied MILP method based on the [47,48].

Our MILP model shows the longest impossible differential characteristics reach 6 rounds. The details of one of these characteristics can be seen in Table C.11. Note that in Table C.11, the "Input differential", "Middle differential", and "Output differential" shows the differentials before S-box layer, after S-box layer, and after linear layers, respectively. Also, the bits "0", "1", and "?" shows zero, active, and unknown differentials, respectively. To prove the impossibility of this differential, we use the following property that can be derived from the Differential Distribution Table (DDT) of $\chi per^{\mathcal{R}}(\cdot)$ S-box.

Fact 1. *The S-box of $\chi per^{\mathcal{R}}(\cdot)$ has the following property:*

- *If the input difference of the S-box is $0x1 = 001$, $0x2 = 010$, and $0x3 = 100$, then the output difference must be as ??1, ?1?, and 1??, respectively, where the ? shows an unknown difference bit.*

C.3. Zero-correlation linear approximation

The zero-correlation attack is one of the cryptanalytic method introduced by Bogdanov and Rijmen [35]. The attack is based on linear approximations with zero correlation. To search for zero-correlation linear approximations, we applied the MILP method [47,48]. The longest

$\chi per^R(A,B)$ $\quad //A=(a^0||a^1||a^2),\ B=(b^0||b^1||b^2),\ |a^i|=|b^i|=n, i=0,1,2.$

{

1. Run expansion algorithm EA as shown in Figure 7 on message $B=(b^0||b^1||b^2)$ to obtain related sub-messages, i.e., $B=(b^0||b^1||b^2) \xrightarrow{EA} \{b_{r,i}\}, r=1,\cdots,\mathcal{R}+1; i=0,1,2.$
2. For $r=1$ to $\mathcal{R}$
   For $i=0$ to 2
   $a^i \leftarrow a^i \oplus b_{r,i}$.
   For $i=0$ to 2 //Non-linear layer
   $v_i = a^i \oplus (a^{(i+1)\%2}\ \&\ \overline{a}^{(i+2)\%2})$.
   Set
   $a^0 \leftarrow v_0 \oplus (v_2 \lll (23\%n))$, //Linear layer
   $a^1 \leftarrow (v_1 \lll 13) \oplus (a^0 \lll 2)$,
   $a^2 \leftarrow (v_2 \lll (19\%n)) \oplus (a^1 \lll 5)$.
3. For $i=0$ to 2
   $z^i \leftarrow a^i \oplus b_{\mathcal{R}+1,i}$.
4. Return $(z^0||z^1||z^2)$.

}

$DM-\chi per^R(A,B)$ $\quad // |A|=|B|=n.$

{

Return $\chi per^R(A,B) \oplus A$.

}

$f_{IV_0}^{DM-\chi per}(M)$ $\quad // M=(m_1||\cdots||m_b).$

{

1. $IV_1 \leftarrow DM-\chi per^R(IV_0, m_1)$.
2. For $i=1$ to $b-1$
   $IV_{i+1} \leftarrow DM-\chi per^R(IV_i, m_{i+1})$.
3. $\tau \leftarrow DM-\chi per^R(IV_b, IV_0)$.
4. Return τ.

}

Fig. B.12. A code-based algorithm of message authentication code $f_{IV_0}^{DM-\chi per}(M)$.

Table C.11
An impossible differential characteristic for 6 rounds $\chi per^R(\cdot)$ when $n=32$.

Round	Input differential	Middle differential	Output differential
↓ 1	00000000000000000000000000000000 00000000000000000000000000000000 00000000000000010000000000000000	00000000000000?0000000000000000 00000000000000?0000000000000000 00000000000000010000000000000000	00000000000000?0000010000000000 00?0000000000?0000010000000000 00000000?0000010000000000000001?00
2	00000000000000?0000010000000000 00?0000000000?0000010000000000 00000000?0000010000000000001?00	00?00000?0000???000?01000000??00 00?00000?0000???00010?000000??00 00?00000?0000?1?000?0?0000001?00	00??0000?0000???000?1?000?0???00 ???00010?00???0???1?0?0?0???0000 01??00???1???1???0?0???000????00
3	00??0000?0000???000?1?000?0???00 ???00010?00???0???1?0?0?0???0000 01??00???1???1???0?0???000????00	????00?????????????????????0?????00 ????00?????????????????????0?????00 ????00???1?????????????????0?????00	

(*continued on next page*)

Table C.11 (*continued*).

Round	Input differential	Middle differential	Output differential
		????????????????0????????????????	???0?????0?????0?????00????0?0??
		????????0????????????????????????	???0?????0?????0?????00????0?0??
		????????0??????????????????????	???0?????0?????0?????00????0?0??
3	???0?????0?????0?????00????0?0??	?0000????0?1???0?????00????000??	?0000??0000010000?0??00?0??0000?
	???0?????0?????0?????00????0?0??	???0????00????00????00010?00?0??	?0000??00000?0000?0??0010??0000?
	???0?????0?????0?????00????0?0??	0??0?1???0?0???000???000???0000?	?0000??00000?0000?0??00?0??0000?
2	?0000??0000010000?0??00?0??0000?	?0000??000001000000000000?00001	000000?000001000000000000000000
	?0000??00000?0000?0??0010??0000?	000000000000000000?0?00010?000000	000000?00000?000000000000000000
	?0000??00000?0000?0??00?0??0000?	000000000000000000000?00001?0000?	000000100000?000000000000000000
↑ 1	000000?000001000000000000000000	000000000000010000000000000000	000000000000000000000000000000
	000000?00000?0000000000000000000	000000000000000000000000000000	000000000000000000000000000000
	0000001000000?0000000000000000000	000000100000000000000000000000	000000000000000000010000000000

Table C.12
A zero-correlation linear approximation for 6 rounds $\chi per^{\mathcal{R}}(\cdot)$ when $n = 32$.

Round	Input mask	Middle mask	Output mask
	00000000000000000000000000000000	0000?0000000000000000000000000	0000?00000000000000?0000000000
↓ 1	00000000000000000000000000000000	0000?0000000000000000000000000	00000000000010000000000??0000000
	00001000000000000000000000000000	00001000000000000000000000000000	00000000000000000100000000000?00
2	0000?0000000000000000?0000000000	0000?0000000?0000?000?0??0000?00	00?0?0?0??00?0?00?000?0??0000?00
	00000000000010000000000??0000000	0000?0000000010000?000?0??0000?00	?000??00?????000??00?00??1000?0?
	00000000000000000100000000000?00	0000?0000000?00001000?0??0000?00	00?0??0000?000?0??0000?00?000?10
3	00?0?0?0??00?0?00?000?0??0000?00	0?0?0???0?????0?0??00??????000???	???0??????????0????0?????????????
	?000??00?????000??00?00??1000?0?	?0?0???0?????0?0??00??????000???	??????!????????0?????????????????
	00?0??0000?000?0??0000?00?000?10	?0?0???0?????0?0??00??????000?1?	0???????0??1????????0?????0?0???
3	??????0??????????????????????????	?0????0?0????0????00????1?0?0??0	?0?0??000?0??0?0??0000??10000100
	??????0??????????????????????????	?00???00?0?0??0?0????0?0??0?0??1	?0?0??000?0?10?0??0000???0000?00
	??????0??????????????????????????	??????0?0????0????0?0????0??????	?0?0??00010??0?0??0000?1?0000?00
2	?0?0??000?0??0?0??0000??10000100	00?0?0000000000?0000?010000100	0000?0000000000000000010000000
	?0?0??000?0?10?0??0000???0000?00	?0000?0000001000?00000000000000	0000?000000000000000000?0000000
	?0?0??00010??0?0??0000?1?0000?00	00?0100001 0?00?0?000001000 0?00	00001000000000000000000?0000000
↑ 1	0000?0000000000000000000010000000	00000000000000000000000010000000	00000000000000000000000010000000
	0000?000000000000000000?0000000	00000000000000000000000000000000	00000000000000000000000000000000
	00001000000000000000000?0000000	00001000000000000000000000000000	00000000000000000000000000000000

zero-correlation linear approximation was obtained for 6 rounds of $\chi per^{\mathcal{R}}(\cdot)$ when $n = 32$. Table C.12 shows an examples of this zero-correlation linear approximation. Note that in this table, the "Input mask", "Middle mask", and "Output mask" shows the linear masks before S-box layer, after S-box layer, and after linear layers, respectively. Also, the bits "0", "1", and "?" shows zero, active, and unknown masks, respectively.

In the same way with impossible differential characteristics, Fact 1 is also true in linear mode and we have used it in Table C.12.

References

[1] BIS. Internet of things market analysis 2020–2027. 2019, https://www.fortunebusinessinsights.com/industry-reports/internet-of-things-iot-market-100307. Last [Accessed 29 march 2021].

[2] research DBM. Global internet of things (IoT) security market – industry trends and forecast to 2026. 2019, https://www.databridgemarketresearch.com/reports/global-internet-of-things-iot-security-market. Last [Accessed on 23 March 2020].

[3] E. L, B. M. Ten trends shaping the internet of things business landscape. 2019, https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights. Last [Accessed 26 April 2019].

[4] Networks PA. 2020 Unit 42 IoT threat report. 2020, https://unit42.paloaltonetworks.com/iot-threat-report-2020/. Last [Accessed on 23 March 2020].

[5] Masmoudi F, Maamar Z, Sellami M, Awad AI, Burégio V. A guiding framework for vetting the internet of things. J Inf Secur Appl 2020;55:102644.

[6] Ammar M, Russello G, Crispo B. Internet of things: A survey on the security of IoT frameworks. J Inf Secur Appl 2018;38:8–27.

[7] El-hajj M, Fadlallah A, Chamoun M, Serhrouchni A. A survey of internet of things (IoT) authentication schemes. Sensors 2019;19(5):1141.

[8] Eldefrawy MH, Pereira N, Gidlund M. Key distribution protocol for industrial internet of things without implicit certificates. IEEE Internet Things J 2018;6(1):906–17.

[9] Tian Q, Lin Y, Guo X, Wen J, Fang Y, et al. New security mechanisms of high-reliability IoT communication based on radio frequency fingerprint. IEEE Internet Things J 2019;6(5):7980–7.

[10] EPCglobal-Inc. EPC radio-frequency identity protocols gen-2 UHF RFID; Specification for RFID air interface protocol. 2015, https://www.gs1.org/sites/default/files/docs/epc/Gen2_Protocol_Standard.pdf. Last [Accessed 26 April 2019].

[11] Peris-Lopez P, Hernandez-Castro JC, Estevez-Tapiador JM, Ribagorda A. LMAP: A real lightweight mutual authentication protocol for low-cost RFID tags. In: Proceedings of RFIDSec06, Austria. 2006.

[12] Peris-Lopez P, Castro JCH, Estévez-Tapiador JM, Ribagorda A. EMAP: An efficient mutual-authentication protocol for low-cost RFID tags. In: Meersman R, Tari Z, Herrero P, editors. OTM workshops, vol. 1. Lecture notes in computer science, vol. 4277, Springer; 2006, p. 352–61.

[13] Tewari A, Gupta BB. Cryptanalysis of a novel ultra-lightweight mutual authentication protocol for IoT devices using RFID tags. J Supercomput 2017;73(3):1085–102.

[14] D'Arco P, Santis AD. On ultralightweight RFID authentication protocols. IEEE Trans Dependable Sec Comput 2011;8(4):548–63.

[15] Phan RC-W. Cryptanalysis of a new ultralightweight RFID authentication protocol - SASI. IEEE Trans Dependable Secure Comput 2009;6(4):316–20.

[16] Tian Y, Chen G, Li J. A new ultra-lightweight RFID authentication protocol with permutation. IEEE Commun Lett 2012;16(5):702–5.

[17] Ahmadian Z, Salmasizadeh M, Aref MR. Desynchronization attack on RAPP ultra-lightweight auth. protocol. Inf Process Lett 2013;113(7):205–9.

[18] Wang S-H, Han Z, Liu S, Chen D-W. Security analysis of RAPP: An RFID authentication protocol based on permutation. 2012, Cryptology ePrint Archive, Report 2012/327.

[19] Bagheri N, Safkhani M, Peris-Lopez P, Tapiador JE. Weaknesses in a new ultra-lightweight RFID authentication protocol with per. - RAPP. Secur Comm Netw 2014;7(6):945–9.

[20] Zhuang X, Zhu Y, Chang C. A new ultra-lightweight RFID protocol for low-cost tags: R ^{2} AP. Wirel Personal Commun 2014;79(3):1787–802.

[21] Khokhar UM, Najam-ul-Islam M, Shami MA. RCIA: A new ultralightweight RFID authentication protocol using recursive hash. IJDSN 2015;2015:642180:1–8.

[22] Mujahid U, Najam-ul Islam M, Sarwar S. A new ultralightweight RFID authentication protocol for passive low cost tags: KMAP. Wirel Pers Commun 2016;1–20.

[23] Luo H, Wen G, Su J, Huang Z. SLAP: Succinct and lightweight authentication protocol for low-cost RFID system. Wirel Netw 2016;1–10.
[24] Khokhar UM, Najam-ul-Islam M, Jafri AR, Qurat-ul-Ain QU, Shami MA. A new ultralightweight RFID mutual authentication protocol: SASI using recursive hash. IJDSN 2016;2016:9648971:1–9648971:14.
[25] Liu Y, Ezerman M, Wang H. Double verification protocol via secret sharing for low-cost RFID tags. Future Gener Comput Syst 2019;90:118–28.
[26] Safkhani M, Bagheri N. Generalized desynchronization attack on UMAP: application to RCIA, KMAP, SLAP and SASI$^+$ protocols, 2016. 2016, p. 905, http://eprint.iacr.org/2016/905.
[27] Damgård IB. A design principle for hash functions. In: Conference on the theory and application of cryptology. Springer; 1989, p. 416–27.
[28] Adeli M, Bagheri N, Sadeghi S, Kumari S. (χ)Perbp: a cloud-based lightweight mutual authentication protocol. 2021, https://eprint.iacr.org/2021/144.
[29] Bertoni G, Daemen J, Peeters M, Assche G. Keccak sponge function family main document (version 2.1), 2010. 2010, Submission to NIST http://keccak.noekeon.org/keccak-main-2.1.pdf.
[30] Biham E, Shamir A. Differential cryptanalysis of DES-like cryptosystems. J Cryptology 1991;4(1):3–72.
[31] Biham E. New types of cryptanalytic attacks using related keys. J Cryptol 1994;7(4):229–46.
[32] Matsui M. Linear cryptanalysis method for DES cipher. In: Workshop on the theory and application of of cryptographic techniques. Springer; 1993, p. 386–97.
[33] Knudsen L. DEAL-a 128-bit block cipher. Complexity 1998;258(2):216.
[34] Biham E, Biryukov A, Shamir A. Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In: International conference on the theory and applications of cryptographic techniques. Springer; 1999, p. 12–23.
[35] Bogdanov A, Rijmen V. Linear hulls with correlation zero and linear cryptanalysis of block ciphers. Des Codes Cryptogr 2014;70(3):369–83.
[36] Gong L, Needham R, Yahalom R. Reasoning about belief in cryptographic protocols. In: Research in security and privacy, IEEE computer society symposium on. IEEE; 1990, p. 234–48.
[37] Burrows M, Abadi M, Needham R. BAN a logic of authentication. Technical report 39, Palo Alto, California: Digital Equipment Systems Research center; 1989.
[38] Blanchet B, Chaudhuri A. Automated formal analysis of a protocol for secure file sharing on untrusted storage. In: Security and privacy, IEEE symposium on. IEEE; 2008, p. 417–31.
[39] Cremers CJF. The Scyther tool: Verification, falsification, and analysis of security protocols. In: Computer aided verification. Springer Berlin; 2008, p. 414–8.
[40] Good T, Benaissa M. Hardware performance of estream phase-III stream cipher candidates. In: Proc. of Workshop on the State of the Art of Stream Ciphers. 2008.
[41] Diehl W, Farahmand F, Yalla P, Kaps J, Gaj K. Comparison of hardware and software implementations of selected lightweight block ciphers. In: 2017 27th international conference on field programmable logic and applications. 2017, p. 1–4.
[42] Jungk B, Apfelbeck J. Area-efficient FPGA implementations of the SHA-3 finalists. In: 2011 international conference on reconfigurable computing and FPGAs. 2011, p. 235–41.
[43] GS1-EPCglobal. EPC tag data standard. 2017, https://www.gs1.org/sites/default/files/docs/epc/GS1_EPC_TDS_i1_11.pdf.
[44] Wu S, Wang M. Security evaluation against differential cryptanalysis for block cipher structures, 2011. 2011, p. 551, IACR Cryptology ePrint Archive.
[45] Mouha N, Wang Q, Gu D, Preneel B. Differential and linear cryptanalysis using mixed-integer linear programming. In: International conference on information security and cryptology. Springer; 2011, p. 57–76.
[46] Sun S, Hu L, Wang P, Qiao K, Ma X, Song L. Automatic security evaluation and (related-key) differential characteristic search: application to SIMON, PRESENT, LBlock, DES (L) and other bit-oriented block ciphers. In: International conference on the theory and application of cryptology and information security. Springer; 2014, p. 158–78.
[47] Cui T, Jia K, Fu K, Chen S, Wang M. New automatic search tool for impossible differentials and zero-correlation linear approximations, 2016. 2016, p. 689, IACR Cryptology ePrint Archive.
[48] Sasaki Y, Todo Y. New impossible differential search tool from design and cryptanalysis aspects. In: Annual international conference on the theory and applications of cryptographic techniques. Springer; 2017, p. 185–215.