




Article

IoT Device Security: Challenging “A Lightweight RFID Mutual Authentication Protocol Based on Physical Unclonable Function”

Ygal Bendavid ^{1,*†}, Nasour Bagheri ^{2,3,†} , Masoumeh Safkhani ^{4,†}  and Samad Rostampour ^{1,5,†} 

¹ Department of Management and Technology, Université du Québec à Montréal (UQAM), Montreal, QC H2X 1L7, Canada; samad.rostampour@iauhvaz.ac.ir

² Electrical Engineering Department, Shahid Rajaei Teacher Training University, Tehran 16788-15811, Iran; NBagheri@sru.ac.ir

³ School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran 19538-33511, Iran

⁴ Computer Engineering Department, Shahid Rajaei Teacher Training University, Tehran 16788-15811, Iran; Safkhani@sru.ac.ir

⁵ Department of Computer Engineering, Ahvaz Branch, Islamic Azad University, Ahvaz 61349-37333, Iran

* Correspondence: bendavid.ygal@uqam.ca

† These authors contributed equally to this work.

Received: 18 October 2018; Accepted: 11 December 2018; Published: 15 December 2018



Abstract: With the exponential increase of Internet of things (IoT) connected devices, important security risks are raised as any device could be used as an attack channel. This preoccupation is particularly important with devices featuring limited processing power and memory capabilities for security purposes. In line with this idea, Xu et al. (2018) proposed a lightweight Radio Frequency Identification (RFID) mutual authentication protocol based on Physical Unclonable Function (PUF)—ensuring mutual tag-reader verification and preventing clone attacks. While Xu et al. claim that their security protocol is efficient to protect RFID systems, we found it still vulnerable to a desynchronization attack and to a secret disclosure attack. Hence, guidelines for the improvements to the protocol are also suggested, for instance by changing the structure of the messages to avoid trivial attacks. In addition, we provide an explicit protocol for which our formal and informal security analysis have found no weaknesses.

Keywords: IoT; RFID; security; physical unclonable function; authentication protocol; desynchronization attack

1. Introduction

The Internet of things (IoT) concept finds its roots in the early 1990s with the vision of ubiquitous computing [1] and the underlying idea that any object can be equipped with technology to become a computing device. This path to digital transformation is now possible as any object can communicate electronically and interact autonomously in real time, with its environment. In fact, the IoT landscape is rapidly changing with an increasing number of vertical applications moving “from experimentation to business scale” [2].

As the IoT market opens up multiple business opportunities, it also introduces a wealth of problems among which security issues usually top the list. This concern will keep increasing exponentially, as the worldwide number of connected devices is expected to jump by 12% on average annually, up to 125 billion connected devices in 2030 [3]. This will lead to important security issues as the ecosystem of connected devices is getting more complex and fragmented. Accordingly, experts anticipate that “by 2020, more than 25% of identified attacks in enterprises will involve the IoT,

although the IoT will account for less than 10% of IT security budgets" [4]. The authors raise the fact that the IoT introduces new challenges (e.g., complexity in solution design architecture, implementation process and integration) as well as a wide range of new security risks "to the IoT devices themselves, their platforms and operating systems, their communications and even the systems to which they're connected". For instance, they suggest that any IoT device could be used as an attack channel.

Therefore, understanding how to build low-cost-high-security IoT devices becomes critical. This awareness is of a particular importance since IoT devices have become "so seamlessly integrated into everyday life, that often the end users are completely unaware of the presence of these devices around them" [5]. This is especially true for low-cost connecting devices in the IoT ecosystem, such as (battery-less) passive Radio-Frequency Identification (RFID) tags, that are powered by the interrogator's radio waves, i.e., by transmitting their signal upon request, when interrogated by a reader. For instance, while passive Ultra High Frequency (UHF) tags are increasingly used in a variety of applications (e.g., identifying products and assets in logistic applications as well as participants in event management), these tags feature limited processing power and memory capabilities for security purposes (i.e., 16-bit cyclic-redundancy check (CRC)) [6].

In a report from the NIST (National Institute of Standards and Technology) [7], the authors present specific recommendations to address potential RFID security risks, including firewalls that separate RFID databases, encryption of radio signals, authentication of approved users, shielding of a reading zone to prevent unauthorized access, logging and time stamping to help in detecting security breaches, etc. Another way of securing RFID tags to support authentication applications is to create hardware security at the silicon level, by using a unique digital signature for RFID silicon chips based on how electrons flow through different paths of the chip, creating a silicon "fingerprint". This approach, firstly introduced by Gassend et al. [8] at the Computer and Communication Security Conference in 2002, was put into practice in 2008 [9], when Verayo, an MIT spinout company, proposed "unclonable" HF tags which included small electronic circuits called PUFs (Physically Unclonable Functions). PUFs take advantage of variations in how silicon Integrated Circuits (ICs) are produced to create a unique digital signature for each chip, similar to biometrics measures, where the "identifiers" cannot be cloned. This allows a challenge–response authentication process that takes into account each tag unique identity based on its IC/transistor random electric properties.

While using chips "digital fingerprint" to build more secure tags, not all proposed PUFs are "unclonable". For instance, in a previous paper entitled "A Lightweight RFID Mutual Authentication Protocol Based on Physical Unclonable Function", Xu et al. [10] examined Kulseng et al.'s [11] proposal of a lightweight mutual verification protocol for RFID systems. Xu et al. found shortcomings in this protocol, making it vulnerable for a desynchronization attack. They hence proposed a security protocol for ensuring mutual tag-reader verification and preventing clone attacks. While the authors claim that their three-stage protocol (tag recognition, mutual verification and update) is efficient to protect RFID systems, we found that Xu et al.'s protocol is still vulnerable to a desynchronization attack and to a secret disclosure attack since the mutual verification process has flaws. These topics and arguments are addressed in this article.

The structure of this paper is as follows: the IoT and related security issues are raised in Section 2. RFID authentication protocols using PUF are discussed in Section 3, and in Section 4 Xu et al.'s protocol will be described. In Section 5, we will then evaluate the security level of the protocol and show its vulnerability. After that, we will present an improved protocol in Section 6 and finally we will conclude in Section 7.

2. The Internet of Things: Infrastructure and Security Issues

The International Telecommunication Union (ITU) defines the IoT as "a global infrastructure enabling advanced services by interconnecting physical and virtual things (i.e., living and non-living entities) based on existing and evolving interoperable information and communication technologies" [12]. The IoT, as a new technological paradigm that aims "to connect anything and anyone

at any time and any place”, [13], is a vision becoming reality with burgeoning IoT enabled applications which open up to multiple business opportunities [14] and lead to new business models [15].

Figure 1 illustrates the IoT reference model ([16]) comprised of seven levels, starting with Level 1: physical devices and controllers that might control multiple devices equipped with various technologies such as Radio Frequency Identification (RFID) tags, Bluetooth Low Energy (BLE) devices, Ultra Sound Identifications (USID), etc. Level 2 represents the communications and connectivity between devices and across networks. Level 3, called Edge (Fog) Computing, allows the conversion of network data flows into suitable information “for storage and higher level processing”. This happens at Level 4 (data accumulation), where event-based data is converted into query-based processing. Level 5 is then required to facilitate the integration with back end systems and “render data and its storage in ways that enable developing simpler, performance-enhanced applications”. It is at Level 6 (Application) that information interpretation takes place to automate or support decision-making in vertical markets’ applications. Level 7 hence represents empowered people and business processes using IoT enabled information to trigger action.

Since security issues can be raised for each level and for the movement of data between levels (e.g., physical cloning at Level 1, unauthorized access at Levels 2 and 3, data manipulation at Levels 4 and 5, and denial of service at Level 6), it goes without saying that security measures must pervade the entire model by (i) securing each device or system (ii) providing security for all processes at each level. Indeed, IoT security requires a multi-level approach.

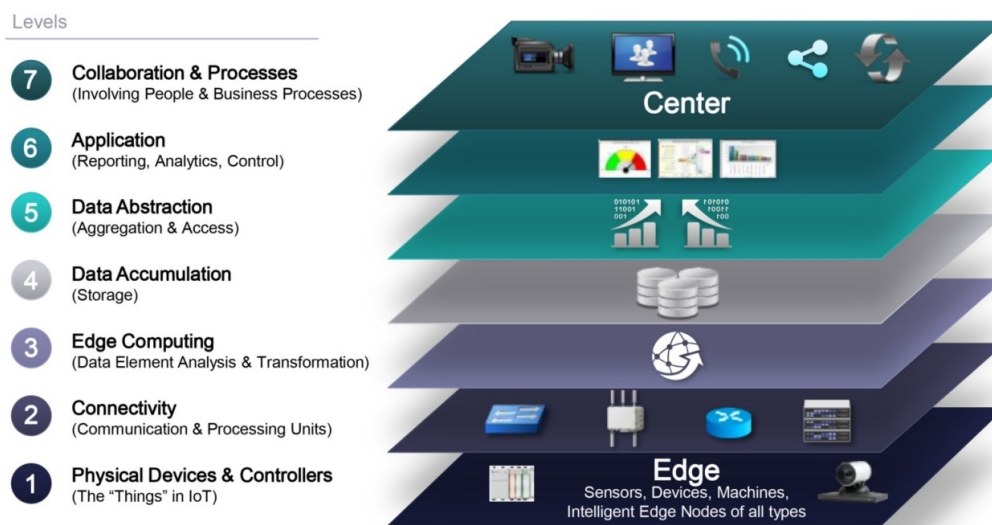


Figure 1. IoT reference model ([16]).

3. RFID Authentication Protocols Using PUF

In this paper, we will focus on level of one of the IoT Reference Model, a level referred by Qi et al. [17] as “the sensing layer” or by Bendavid [18] as “the data capture layer”. At this level, one method to increase the security and to prohibit unauthorized access in IoT applications is to perform tag-reader mutual authentication protocol [19–22]. Various security techniques can be applied in order to transfer data securely such as cryptosystems. These cryptosystems are commonly divided into two groups: one-way and two-way [23]. In one-way systems, data is encrypted and there is no module for decrypting, such as Hash or PUF. In two-way systems, data is encrypted and decrypted by a cryptographic algorithm. Furthermore, within one-way systems, Hash, as a function, does not have the capability to detect physical attacks. On the other hand, PUF has an unclonable function that utilizes physical features to improve the security and resist against physical threats.

In recent years, many researchers have proposed more secure authentication protocols based on PUF technology [23,24]. For example, Sadeghi et al. presented an efficient privacy-preserving protocol based on rooted on PUF and claimed that their protocol was cost-efficient and provided an acceptable level of security and privacy [25]. However, despite Sadeghi et al.'s claim, Kardas et al. proved that the recommended protocol was vulnerable to the impersonation, as an attacker could trace the communication of a tag [26]. In addition, Kardas et al. analyzed another PUF-based authentication protocol that has been endorsed by Akgun et al. [27] and showed that the protocol was vulnerable and could not guarantee the security of the system [26]. Akgun et al. presented another authentication protocol by using PUF technology, but their protocol was still to be found insecure and could not provide forward secrecy support [28]. In order to solve the controversy surrounding these issues, Kardas et al. also presented another PUF-based protocol that was unfortunately not resistant against Denial of Service (DoS) attacks [29].

While most PUF based protocols were initially planned for an ideal environment with limited applicability in real situations, some researchers proposed authentication protocols for real and noisy environments [30–33]. For example, Ray et al. [34] released a PUF based protocol for object tracking in IoT application in real environments. In addition, Huth et al. proposed a protocol that can resist against noisy environment [33]. Nevertheless, their protocol needs an extra channel for transferring data and they use an exhaustive search in the database to retrieve the tag's information. However, in their defense, most PUF protocols suffer the extra load of exhaustive search.

Recently, Kulseng et al. also disclosed a lightweight mutual verification protocol [11]. In order to implement a lightweight protocol, they utilized a PUF and Linear Feedback Shift Register (LFSR) instead of complex functions. However, while the proposed protocol was suitable for low-cost and large-scale systems, it could not comply with the security requirement of RFID systems. Xu et al. analyzed Kulseng et al.'s protocol and proved that it was vulnerable to the desynchronization attack and data confidentiality [10]. The authors presented a new PUF-based protocol and claimed that this improved protocol solved the security issues of the previous one and was robust against RFID threats. Before demonstrating its weaknesses, we will elaborate on Xu et al.'s protocol in more detail in Section 4.

4. Xu et al.'s Protocol

In this section, we explain the PUF-based lightweight authentication protocol that has been proposed by Xu et al. The order of transferred messages is illustrated in Figure 2 and the used notations are listed in Table 1.

Table 1. The notations used in the Xu et al.'s protocol.

Notations	
FID	Fake tag ID
$PUF()$	Physical Unclonable Function
ID	The tag's ID number
TID	Pseudo ID of the tag
P_n	The tag's secret value
K_n	The shared secret value of the tag and the reader
\lll	The left rotation operator
$\&$	The AND operator
\parallel	The concatenation operator
r_i	A random number
\oplus	XOR function
\mathcal{X}_i	Denotes the i th bit of the string \mathcal{X} , where \mathcal{X}_0 is the most right bit

This protocol consists of four phases: Registration, Tag verification, Mutual verification and Update process. In the registration phase, the initial values of the parameters are stored in the tags and in the database. The database stores two sets of values for each tag: the current values

$(FID^{new}, P_n^{new}, P_{n+1}^{new}, K_n^{new})$ and the old values $(FID^{old}, P_n^{old}, P_{n+1}^{old}, K_n^{old}, ID)$. If an authentication process is done completely and a tag updates its parameters successfully, the database uses the current values; otherwise, it will use the old values.

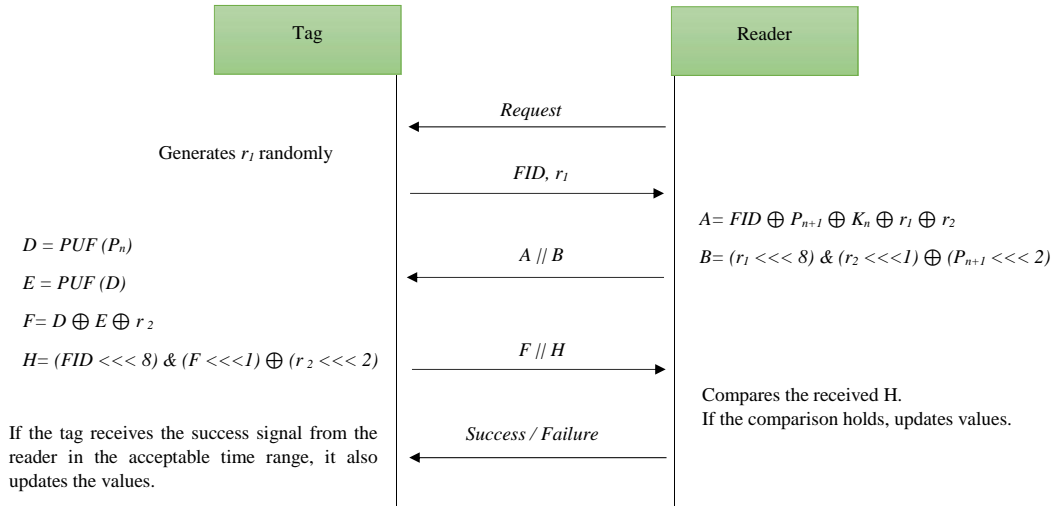


Figure 2. Time sequence diagram of Xu et al.'s protocol for single tag authentication.

4.1. Tag Verification Phase

At the first of verification phase, the reader starts the process and sends a search request to the tag and receives the tag's response as FID and r_1 . The reader looks up FID in the new fields and old fields. Finally, if the appropriate FID is not found, the reader terminates the session; otherwise, it continues the process.

4.2. Mutual Verification Phase

In this phase, the reader generates r_2 randomly and computes A and B as:

$$A = FID \oplus P_{n+1} \oplus K_n \oplus r_1 \oplus r_2, \tag{1}$$

$$B = (r_1 \lll 8) \& (r_2 \lll 1) \oplus (P_{n+1} \lll 2). \tag{2}$$

It concatenates A and B and transfers the combination message $(A || B)$ to the tag. Upon receiving the message from the reader, the tag computes D and E as follows:

$$D = PUF(P_n), \tag{3}$$

$$E = PUF(D). \tag{4}$$

In addition, it calculates r'_2 by the receiving messages to recalculate B' as:

$$r'_2 = A \oplus FID \oplus D \oplus K_n \oplus r_1, \tag{5}$$

$$B' = (r_1 \lll 8) \& (r'_2 \lll 1) \oplus (D \lll 2). \tag{6}$$

The tag checks $B = B'$. If the equation does not hold, it means that the reader is not valid and the session is terminated. Otherwise, the tag verifies the reader and continues the process.

Based on D, E and calculated r'_2 , the tag computes F and then H as:

$$F = D \oplus E \oplus r'_2, \tag{7}$$

$$H = (FID \lll 8) \& (F \lll 1) \oplus (r'_2 \lll 2). \quad (8)$$

It concatenates F and H as $(F||H)$ and sends this message to the reader.

When the reader receives the tag's response, by local parameters, it recalculates E' , F' and H' as:

$$E' = F \oplus r_2 \oplus P_{n+1}, \quad (9)$$

$$F' = E' \oplus r_2 \oplus P_{n+1}, \quad (10)$$

$$H = (FID \lll 8) \& (F' \lll 1) \oplus (r_2 \lll 2). \quad (11)$$

Then, it checks $H = H'$. If the equation does not hold, the tag is not valid and the reader aborts the session. Otherwise, it verifies the tag and continues the process.

4.3. Update Process Phase

After the verification phase, the variables should be updated. At first, the database updates its parameters, and then it sends a signal to the tag to inform it that the update process was done. If the tag receives the signal in the acceptable time range, it also updates its local parameters. Before updating in the database, the reader checks FID in the tag verification phase. If it has used FID^{old} , the reader does not need to update the parameters, but if FID^{new} has been used, the reader updates the database as:

$$P_n^{new} = P_{n+1}^{new}, \quad P_{n+1}^{new} = E', \quad (12)$$

$$K_n^{new} = (K_n^{new} \lll 8) \& (r'_2 \lll 1) \oplus (r_1 \lll 2), \quad (13)$$

$$FID^{new} = (FID^{new} \lll 8) \& (r'_2 \lll 1) \oplus (r_1 \lll 2), \quad (14)$$

$$FID^{old} = FID^{new}, P_n^{old} = P_n^{new}, P_{n+1}^{old} = P_{n+1}^{new}, K_n^{old} = K_n^{new}. \quad (15)$$

If the tag receives a message (positive signal) from the reader in the acceptable time range, it also updates the parameters as:

$$P_n = D, \quad (16)$$

$$FID = (FID \lll 8) \& (r'_2 \lll 1) \oplus (r_1 \lll 2), \quad (17)$$

$$K_n = (K_n \lll 8) \& (r'_2 \lll 1) \oplus (r_1 \lll 2). \quad (18)$$

5. Security Analysis of Xu et al.'s Protocol

While Xu et al. claimed that their security protocol is efficient to protect RFID systems, we found that their protocol is still at risk of being exposed to a desynchronization attack and to a secret disclosure attack. In this section, we will explain how these attacks can strongly jeopardize the security of the system.

5.1. Desynchronization Attack

The goal of a desynchronization attack is to change the integrity and synchronization of stored data in the tag and in the reader [35]. The main observation in the proposed attack mentioned in this section is the fact given $A \& B = C$; any modification on a bit of A has no effect on the result C with the probability of $\frac{1}{2}$. More precisely, we assume that $B_i = 0$, where B_i denotes the i th bit of the string B , $A_i \& B_i = 0$, for any value of $A_i \in \{0, 1\}$. Following this observation, the attack procedure, which is a desynchronization attack when the adversary is modeled as a man in the middle, works as presented in the following steps:

1. On a legitimate session between the target tag T and the reader R , the reader starts the process and sends a search request to the tag.

2. Upon receiving the request, the tag generates r_1 randomly and transfers FID and r_1 to the reader.
3. The adversary intercepts the sent message from the tag to the reader and replaces r_1 with $r_1 \oplus 0x1$.
4. When the reader receives FID , it utilizes it as a search key in the database and seeks to find an equal FID in the FID^{new} field. We assume that the comparison holds, meaning that the tag belongs to the system and the reader uses $FID^{new}, P_n^{new}, P_{n+1}^{new}, K_n^{new}$ for the next phase.
5. The reader generates r_2 randomly and computes A and B as:

$$A = FID \oplus P_{n+1} \oplus K_n \oplus (r_1 \oplus 0x1) \oplus r_2, \quad (19)$$

$$B = ((r_1 \oplus 0x1) \lll 8) \& (r_2 \lll 1) \oplus (P_{n+1} \lll 2). \quad (20)$$

6. It concatenates A and B and transfers the combination message ($A||B$) to the tag.
7. The adversary intercepts the sent message from the reader to the tag and replaces A with $A \oplus 0x1$.
8. Upon receiving the message from the reader, the tags computes D and E as follows:

$$D = PUF(P_n), \quad (21)$$

$$E = PUF(D). \quad (22)$$

9. In addition, it calculates r'_2 by the receiving messages to recalculate B' as:

$$r'_2 = A \oplus FID \oplus D \oplus K_n \oplus r_1, \quad (23)$$

$$B' = (r_1 \lll 8) \& (r'_2 \lll 1) \oplus (D \lll 2). \quad (24)$$

10. The tag checks $B = B'$. If the equation does not hold, it means the reader is not valid and the session is terminated. Otherwise, the tag verifies the reader and continues the process.
11. Based on D, E and calculated r'_2 , the tag computes F and then H as:

$$F = D \oplus E \oplus r'_2, \quad (25)$$

$$H = (FID \lll 8) \& (F \lll 1) \oplus (r'_2 \lll 2). \quad (26)$$

12. It concatenates F and H as ($F||H$) and sends this message to the reader.
13. When the reader receives the tag's response, by local parameters, it recalculates E', F' and H' as:

$$E' = F \oplus r_2 \oplus P_{n+1}, \quad (27)$$

$$F' = E' \oplus r_2 \oplus P_{n+1}, \quad (28)$$

$$H = (FID \lll 8) \& (F' \lll 1) \oplus (r_2 \lll 2). \quad (29)$$

14. Then, it checks $H = H'$. If it does not hold, the tag is not valid, and the reader aborts the session. Otherwise, it verifies the tag and continues the process.
15. Based on the assumption in Step 4, in the update process phase of the protocol verification phase, the database updates its parameters and then sends a signal to the tag to inform it that the update process is completed. The reader updates the database as follows:

$$P_n^{new} = P_{n+1}^{new} \quad , \quad P_{n+1}^{new} = E', \quad (30)$$

$$K_n^{new} = (K_n^{new} \lll 8) \& (r'_2 \lll 1) \oplus ((r_1 \oplus 1) \lll 2), \quad (31)$$

$$FID^{new} = (FID^{new} \lll 8) \& (r'_2 \lll 1) \oplus ((r_1 \oplus 1) \lll 2), \quad (32)$$

$$FID^{old} = FID^{new}, P_n^{old} = P_n^{new}, P_{n+1}^{old} = P_{n+1}^{new}, K_n^{old} = K_n^{new}. \quad (33)$$

16. When the tag receives the success signal from the reader in the acceptable time range, it also updates the parameters as:

$$P_n = D, \quad (34)$$

$$FID = (FID \lll 8) \& (r'_2 \lll 1) \oplus (r_1 \lll 2), \quad (35)$$

$$K_n = (K_n \lll 8) \& (r'_2 \lll 1) \oplus (r_1 \lll 2). \quad (36)$$

It is clear that the above attack succeeds if:

$$(r_1 \lll 8) \& (r_2 \lll 1) \oplus (P_{n+1} \lll 2) = ((r_1 \oplus 0x1) \lll 8) \& (r_2 \lll 1) \oplus (P_{n+1} \lll 2). \quad (37)$$

In light of these findings, the probability of the above equality is $\frac{1}{2}$. Given that the only impact of r_1 is on the calculation of $A\|B$, if the above equality is satisfied, T and R authenticate each other successfully and the adversary is not recognized. However, the tag updates its values with r_1 while the reader uses $r_1 \oplus 1$ in the update process. Hence, the tag's records of secret parameters do not match the records of the reader and they will be desynchronized with the probability of almost $\frac{1}{2}$, while the complexity of the attack method is just intercepting two steps of a session of the protocol between T and R .

5.2. Secret Disclosure Attack

The main observation which is used in the proposed secret disclosure attack is the fact that $A_i \& 0 = 0$ and $A_i \& 1 = A_i$, for any value of $A_i \in \{0, 1\}$. Following this observation, the attack procedure, when the adversary is modeled as a man in the middle, works as follows:

1. On a legitimate session between the target tag T and the reader R , the reader starts the process and sends a search request to the tag.
2. Upon receiving the request, the tag generates r_1 randomly and transfers FID and r_1 to the reader.
3. The adversary intercepts the sent message from the tag to the reader and just replaces r_1 by 0 and stores FID .
4. When the reader receives FID , it utilizes it as a search key in the database and seeks to find an equal FID in the FID^{new} field. We assume that the comparison holds, meaning that the tag belongs to the system and the reader uses $FID^{new}, P_n^{new}, P_{n+1}^{new}, K_n^{new}$ for the next phase.
5. The reader generates r_2 randomly and computes A and B as:

$$A = FID \oplus P_{n+1} \oplus K_n \oplus (0) \oplus r_2, \quad (38)$$

$$B = (0 \lll 8) \& (r_2 \lll 1) \oplus (P_{n+1} \lll 2) = P_{n+1} \lll 2. \quad (39)$$

6. The adversary stores $A\|B$ and blocks it, where $B = P_{n+1} \lll 2$ and P_{n+1} is the tag's secret value and $A = FID \oplus P_{n+1} \oplus K_n \oplus r_2$.
7. The adversary waits until the reader starts the process and sends a search request to a tag.
8. Upon receiving the request, the adversary sets $r'_1 = 0x1 \dots 1$ and transfers the stored FID and r'_1 to the reader.
9. When the reader receives FID , it utilizes it as a search key in the database and seeks to find an equal FID in the FID^{new} field. Based on the assumption of Step 4, the comparison holds, meaning that the tag belongs to the system and the reader uses $FID^{new}, P_n^{new}, P_{n+1}^{new}, K_n^{new}$ for the next phase.
10. The reader generates r'_2 randomly and computes A' and B' as:

$$A' = FID \oplus P_{n+1} \oplus K_n \oplus (0x1 \dots 1) \oplus r'_2, \quad (40)$$

$$B' = ((0x1 \dots 1) \lll 8) \& (r'_2 \lll 1) \oplus (P_{n+1} \lll 2). \quad (41)$$

11. The adversary stores $A' || B'$ and blocks it, where $B' = (r'_2 \lll 1) \oplus (P_{n+1} \lll 2)$. Given P_{n+1} from Step 6, the adversary extracts r'_2 as $r'_2 = ((B' \oplus (P_{n+1} \lll 2)) \ggg 2)$ and K_n as $K_n = A' \oplus FID \oplus P_{n+1} \oplus (0x1 \dots 1) \oplus r_2$.

On the basis of these elements, the probability of success of the adversary to retrieve the tag's secret value is 1, while the complexity of the attack method is just intercepting a step of the sessions of the protocol between T and R and eavesdropping the transferred values over a public channel. It should also be noted that, given P_{n+1} and K_n , the adversary can impersonate the reader and communicate with the tag to extract the rest of the secret parameters as follows:

1. The adversary impersonates the legitimate reader and sends a search request to the target tag.
2. Upon receiving the request, the tag generates r''_1 randomly and transfers FID and r''_1 to the reader (adversary).
3. The adversary generates r''_2 randomly and computes A'' and B'' as follows and sends them to the tag:

$$A'' = FID \oplus P_{n+1} \oplus K_n \oplus r''_1 \oplus r''_2, \quad (42)$$

$$B'' = (r''_1 \lll 8) \& (r''_2 \lll 1) \oplus (P_{n+1} \lll 2). \quad (43)$$

4. Upon receiving the message, the tag computes $D = PUF(P_n)$ and $E = PUF(D)$, verifies the received B'' to validate the reader and continues the process.
5. Based on D, E and calculated r''_2 , the tag computes F and then H as:

$$F'' = D \oplus E \oplus r''_2, \quad (44)$$

$$H'' = (FID \lll 8) \& (F'' \lll 1) \oplus (r''_2 \lll 2). \quad (45)$$

6. It concatenates F'' and H'' as $(F'' || H'')$ and sends this message to the reader (adversary).
7. When the adversary receives the tag's response, it extracts $E = D \oplus F'' \oplus r''_2$, sends a signal to the tag to inform that the update process was done, and does the following updates in the stored information related to the target tag:

$$P_n^{new} = P_{n+1} \quad , \quad P_{n+1}^{new} = E, \quad (46)$$

$$K_n^{new} = (K_n^{new} \lll 8) \& (r''_2 \lll 1) \oplus ((r''_1 \oplus 1) \lll 2), \quad (47)$$

$$FID^{new} = (FID^{new} \lll 8) \& (r''_2 \lll 1) \oplus ((r''_1 \oplus 1) \lll 2). \quad (48)$$

8. When the tag receives the success signal from the reader (adversary) in the acceptable time range, it also updates the parameters as:

$$P_n = D, \quad (49)$$

$$FID = (FID \lll 8) \& (r''_2 \lll 1) \oplus (r''_1 \lll 2), \quad (50)$$

$$K_n = (K_n \lll 8) \& (r''_2 \lll 1) \oplus (r''_1 \lll 2). \quad (51)$$

From now on, the adversary is in possession of all the tags secrets and has also desynchronized the tag and the reader. Hence, it is only the adversary who can communicate with the victim tag. The probability of success of the attack is 1 and the complexity of the attack method is negligible. One may argue that, in order to patch the given secret discourse attack, the reader shall refuse trivial values as r_1 , e.g., it should reject any value which does not sound like it is random such as $r_1 = 0 \dots 0$ and $r_1 = 1 \dots 1$. However, this approach does not rule out the projected attack, although it can slightly increase the complexity of the attack. In this case, the scenario of an attack could be as follows:

1. In a legitimate session between the target tag T and the reader R , the reader starts the process and sends a search request to the tag.
2. Upon receiving the request, the tag generates r_1 randomly and transfers FID and r_1 to the reader.
3. The adversary intercepts the sent message from the tag to the reader and just replaces r_1 by 0 and stores FID .
4. When the reader receives FID , it utilizes it as a search key in the database and seeks to find an equal FID in the FID^{new} field. We assume that the comparison holds, meaning that the tag belongs to the system and the reader uses $FID^{new}, P_n^{new}, P_{n+1}^{new}, K_n^{new}$ for the next phase.
5. The reader generates r_2 randomly and computes A and B as:

$$A = FID \oplus P_{n+1} \oplus K_n \oplus r_1 \oplus r_2, \quad (52)$$

$$B = (r_1 \lll 8) \& (r_2 \lll 1) \oplus (P_{n+1} \lll 2). \quad (53)$$

6. The adversary stores $A||B$ and blocks it, where, for any $(r_1)_i = 0$, we have $B_{i+6} = (P_{n+1})_{i+6}$ and P_{n+1} is the tag's secret value and $A = FID \oplus P_{n+1} \oplus K_n \oplus r_2$.
7. The adversary waits until the reader starts another session of the protocol and sends a search request to a tag.
8. Upon receiving the request, the adversary sets $r'_1 = r_1 \oplus 0x1 \dots 1$ and transfers the stored FID and r'_1 to the reader.
9. When the reader receives FID , it utilizes it as a search key in the database and seeks to find an equal FID in the FID^{new} field. We again assume that the comparison holds, meaning that the tag belongs to the system and the reader uses $FID^{new}, P_n^{new}, P_{n+1}^{new}, K_n^{new}$ for the next phase.
10. The reader generates r'_2 randomly and computes A' and B' as:

$$A' = FID \oplus P_{n+1} \oplus K_n \oplus r'_1 \oplus r'_2, \quad (54)$$

$$B' = (r'_1 \lll 8) \& (r'_2 \lll 1) \oplus (P_{n+1} \lll 2). \quad (55)$$

11. The adversary stores $A'||B'$ and blocks it, where, for any $(r'_1)_i = 0$ $B_{i+6} = (P_{n+1})_{i+6}$. Combining these bits of P_{n+1} with the bits retrieved in Step 6, the adversary has a whole P_{n+1} .
12. The adversary waits until the reader starts the next process and sends a search request to a tag.
13. Upon receiving the request, the adversary generates a random value as r_1 , transfers r_1 and the stored FID to the reader.
14. When the reader receives FID , it utilizes it as a search key in the database and seeks to find an equal FID in the FID^{new} field. Based on the assumption of Step 4, the comparison holds, meaning that the tag belongs to the system and the reader uses $FID^{new}, P_n^{new}, P_{n+1}^{new}, K_n^{new}$ for the next phase.
15. The reader generates r_2 randomly and computes A and B as:

$$A = FID \oplus P_{n+1} \oplus K_n \oplus r_1 \oplus r_2, \quad (56)$$

$$B = ((0x1 \dots 1) \lll 8) \& (r_2 \lll 1) \oplus (P_{n+1} \lll 2). \quad (57)$$

16. The adversary stores $A||B$ and blocks it, where, for any $(r_1)_i = 1$, we have $B_{i+6} = (P_{n+1})_{i+6} \oplus r_1 \oplus (r_2)_{i+7}$ and $A_{i+7} = (FID \oplus P_{n+1} \oplus K_n \oplus r_2)_{i+7}$. Given that the adversary has FID, P_{n+1}, r_1 and $(r_2)_{i+7}$, it can extract $(K_n)_j$ for any j that $(r_1)_{j-7} = 1$.
17. The adversary waits until the reader starts another session of the protocol and sends a search request to a tag.
18. Upon receiving the request, the adversary sets $r'_1 = r_1 \oplus 0x1 \dots 1$ and transfers the stored FID and r'_1 to the reader.

19. When the reader receives FID , it utilizes it as a search key in the database and seeks to find an equal FID in the FID^{new} field. We again assume that the comparison holds, meaning that the tag belongs to the system and the reader uses $FID^{new}, P_n^{new}, P_{n+1}^{new}, K_n^{new}$ for the next phase.
20. The reader generates r'_2 randomly and computes A' and B' as:

$$A' = FID \oplus P_{n+1} \oplus K_n \oplus r'_1 \oplus r'_2, \quad (58)$$

$$B' = (r'_1 \lll 8) \& (r'_2 \lll 1) \oplus (P_{n+1} \lll 2). \quad (59)$$

21. $(r_1)_i = 1$, we have $B'_{i+6} = (P_{n+1})_{i+6} \oplus (r'_2)_{i+7}$ and $A'_{i+7} = (FID \oplus P_{n+1} \oplus K_n \oplus r'_1 \oplus r'_2)_{i+7}$. Given that the adversary has FID, P_{n+1}, r'_1 and $(r'_2)_{i+7}$, it can extract $(K_n)_j$ for any j that $(r'_1)_{j-7} = 1$. Combining these bits of K_n with the bits retrieved in Step 16, the adversary has a whole K_n .

It is clear that, in the given attack, the probability of success of the adversary to retrieve the tag's secret value is 1, while the complexity of the attack method is just intercepting/eavesdropping four sessions of the protocol between T and R . Even if the reader stores the history of all transferred random values between the tag and the reader, it is possible to adapt the given attack to deceive the reader and extract the secret parameters, although we may require a few extra number of sessions. It should be noted that we did not use the tag responses, i.e., F and H , in the proposed secret disclosure attacks. For example, in Xu et al.'s protocol, FID and F are also transferred over a public channel and:

$$F = D \oplus E \oplus r_2, \quad (60)$$

$$H = (FID \lll 8) \& (F \lll 1) \oplus (r_2 \lll 2). \quad (61)$$

Hence, the adversary can easily extract r_2 from the available information. Given r_2 and r_1 and $A \parallel B$, where:

$$A = FID \oplus P_{n+1} \oplus K_n \oplus r_1 \oplus r_2, \quad (62)$$

and

$$B = (r_1 \lll 8) \& (r_2 \lll 1) \oplus (P_{n+1} \lll 2). \quad (63)$$

The adversary can easily extract P_{n+1} and K_n . Given r_2 and $P_{n+1} = D$ and F , the adversary can extract E . Hence, the adversary extracts all secret parameters passively. The probability of success of this attack is also 1. However, it necessarily requires eavesdropping F and H .

6. Improved Protocol

To avoid trivial attacks, it is possible to design a new protocol based on Xu et al.'s protocol by changing the structure of the messages programmed. For instance, any transferred information should be masked. Therefore, it is better to transfer $r_1 \oplus P_n$ instead of the plain r_1 or use ID to generate the new TID rather than the TID itself, which is known by the adversary. Furthermore, considering the fact that the output of AND operation is '0' with the probability of $\frac{3}{4}$, one should avoid it and try to use XOR which is a balanced bitwise operation. However, all things considered, the history of many easily broken ultra-lightweight protocols in literature shows that even such an improved protocol may not be that secure, not to mention that we use enough nonlinear and complicated functions, instead of just a few bitwise operations. Under the circumstances, it could be better to use available well-known lightweight primitives such as SIMON [36], Skinny [37], NORX [38] and SPONGENT [39]. For example, it is possible to implement SPONGENT in an area equal to 738 NAND gate [40]. Based on this observation, we propose in this section an improved authentication protocol apart from PUF, which uses a cryptographic hash function as the core of security. It should be noted many lightweight hash functions have already been introduced in literature, e.g., PHOTON [41], Quark [42] and SPONGENT [39]. A comprehensive comparison of those primitives can be found at CryptoLUX [40].

6.1. The Proposed Protocol

The proposed protocol, as it is depicted in Figure 3, includes two phases: the registration phase over a secure channel and the authentication phase over a public channel.

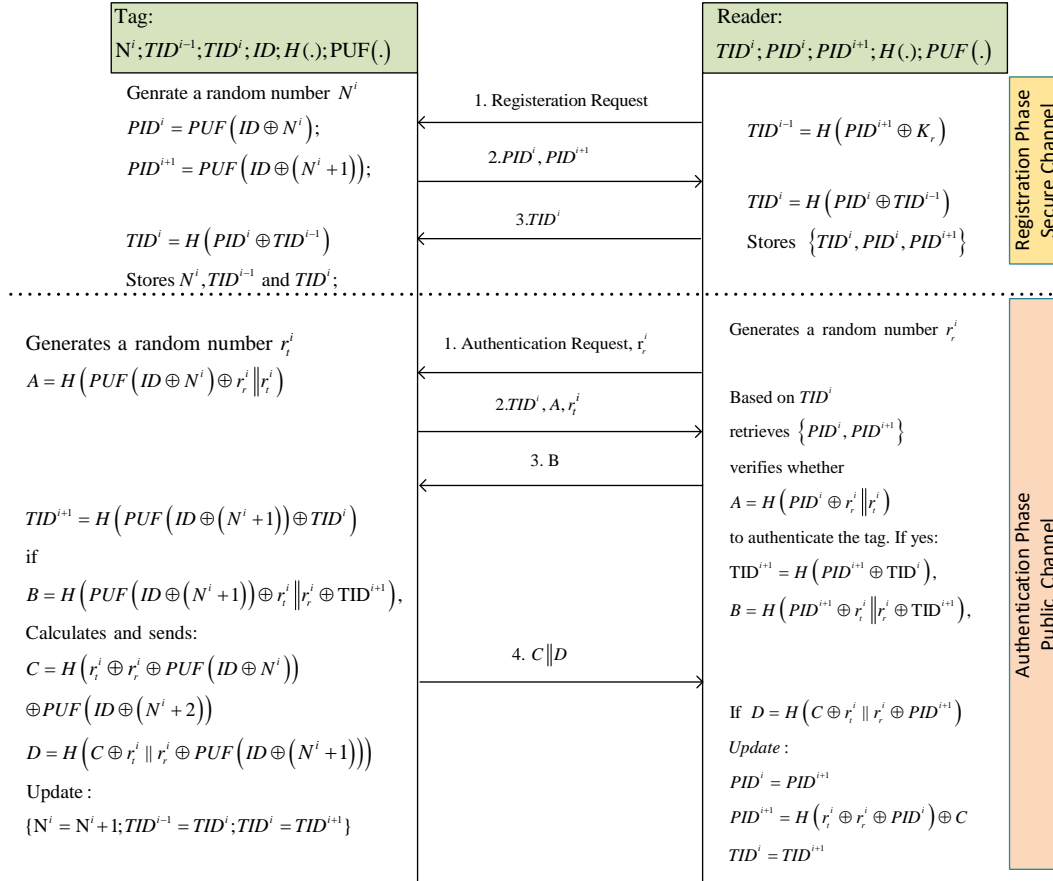


Figure 3. The improved PUF-based protocol.

Registration Phase:

In this phase, the tag T^i is registered into the reader as follows:

1. The reader sends a registration request command to the tag.
2. The tag generates a random number N^i , calculates PID^i and PID^{i+1} as follows and sends them to the reader:

$$PID^i = PUF(ID \oplus N^i),$$

$$PID^{i+1} = PUF(ID \oplus (N^i + 1)).$$

3. The reader calculates a temporary identifier $TID^{i-1} = H(PID^{i+1} \oplus K_r)$ for the tag and sends to it, where K_r is the secret parameter of the reader.
4. It also updates TID^i as $TID^i = H(PID^i \oplus TID^{i-1})$ and creates a record in its database for this tag, including $\{TID^i, PID^i, PID^{i+1}\}$.
5. Given TID^{i-1} , the tag calculates $TID^i = H(PID^i \oplus TID^{i-1})$. It stores $\{N^i, TID^{i-1}, TID^i\}$.

Authentication Phase:

In this phase, the tag T^i and the reader R are mutually authenticated as follows:

1. The reader sends an authentication request command along a random number r_r^i to the tag.
2. The tag generates a random number r_t^i , calculates $A = H(PUF(ID \oplus N^i) \oplus r_r^i || r_t^i)$ and sends tuple (A, TID^i, r_t^i) to the reader.

3. Given TID^i , the reader finds the related record in its database, i.e., $\{TID^i, PID^i, PID^{i+1}\}$. Then, the reader verifies whether $A \stackrel{?}{=} H(PID^i \oplus r_r^i \| r_t^i)$ to authenticate the tag. If the tag is authenticated, then the reader calculates $TID^{i+1} = H(PID^{i+1} \oplus TID^i)$ and $B = H(PID^{i+1} \oplus r_t^i \| r_r^i \oplus TID^{i+1})$. Next, it sends B to the tag.
4. The tag also calculates $TID^{i+1} = H(PUF(ID \oplus (N^i + 1)) \oplus TID^i)$ and verifies whether $B \stackrel{?}{=} H(PUF(ID \oplus (N^i + 1)) \oplus r_t^i \| r_r^i \oplus TID^{i+1})$, to authenticate the reader. If the reader is certified as valid, the tag does the following calculations:

$$\begin{aligned} C &= H(r_t^i \oplus r_r^i \oplus PUF(ID \oplus N^i)) \oplus PUF(ID \oplus (N^i + 2)), \\ D &= H(C \oplus r_t^i \| r_r^i \oplus PUF(ID \oplus (N^i + 1))). \end{aligned}$$

5. Next, the tag sends (C, D) to the reader and updates its records as $\{N^i = N^i + 1; TID^{i-1} = TID^i; TID^i = TID^{i+1}\}$.
6. Once the reader received (C, D) , it verifies whether $D \stackrel{?}{=} H(C \oplus r_t^i \| r_r^i \oplus PID^{i+1})$, to authenticate the tag and the received data. If it is authenticated, then the reader updates its records as follows:

$$\begin{aligned} PID^i &= PID^{i+1}, \\ PID^{i+1} &= H(r_t^i \oplus r_r^i \oplus PID^i) \oplus C, \\ TID^i &= TID^{i+1}. \end{aligned}$$

7. If the tag has not been authenticated based on N^i and TID^i , it assumes that the last session was not finished successfully and the tag tries to be authenticated based on $N^i - 1$ and TID^{i-1} .

6.2. Security Evaluation of the Proposed Protocol

In this section, we evaluate the security of the proposed protocol informally and formally. For this purpose, we considered an active adversary, which is able to eavesdrop on any transferred messages over the public channel at different stages of the procedure, e.g., authentication phase of the proposed protocol, modify/block messages or initiate a session to impersonate the reader/tag. However, we specify that the adversary cannot influence or access the transferred messages over a secured channel, e.g., registration phase of the proposed protocol. In addition, we assume that the embedded PUF behaves randomly from a tag to a tag and on different inputs, i.e., $Pr(PUF^i(x_1) = PUF^i(x_2)) = 2^{-n}$ and $Pr(PUF^i(x_1) = PUF^j(x_1)) = 2^{-n}$, where $x_1 \neq x_2$, n is the output length of PUF^i and PUF^i is the PUF which is integrated in the tag T^i .

6.2.1. Informal Analysis

Traceability Attack: To trace a tag/reader and to compromise the protocol's anonymity, the adversary should be able to link transferred messages between the tag and the reader to their identity or to the previous transferred messages. On the other hand, on each run of the protocol, the transferred messages include $r_r^i, r_t^i, TID^i, A=H(PID^i \oplus r_r^i \| r_t^i), B = H(PID^{i+1} \oplus r_t^i \| r_r^i \oplus TID^{i+1}), C = H(r_t^i \oplus r_r^i \oplus PUF(ID \oplus N^i)) \oplus PUF(ID \oplus (N^i + 2))$ and $D = H(C \oplus r_t^i \| r_r^i \oplus PUF(ID \oplus (N^i + 1)))$, where $TID^{i+1} = H(PID^{i+1} \oplus TID^i)$. r_r^i and r_t^i are fresh random values generated in each session and they cannot be used to trace the tag/reader. TID^i is the pseudo identity of the tag which is updated at the end of each successful run of the protocol. Moreover, the tag keeps its record up to the next successful run of the protocol. Although the record can be used to trace the tag even after one successful run of the protocol, it is impossible to employ it later.

More precisely, an adversary who eavesdropped on a session of the protocol between the tag and the reader and who missed two consecutive successful runs of the protocol after that will not be able to use the eavesdropped value of TID^i to trace the target tag. It should be noted that it is possible to fix this problem by masking the transferred pseudo-identity, for example as $H(TID^{i+1} \oplus r_t^i)$. However,

in doing so, the reader needs to search through its entire database to find the tag that compromises the protocol's scalability. Other solutions, such as encrypting TID by the public key of the reader, are also possible, but it increases the computational complexity of the protocol. The rest of the messages are randomized by random values contributed by both the tag and the reader and a secure hash function. Hence, even an adversary who is impersonating the reader or the tag cannot control the random numbers generated by the other party and cannot predict the transferred messages from one session to another session.

Secret Disclosure Attack: The main secret parameters that are shared with the reader are PID^i and PID^{i+1} . They are used in generating different messages that are transferred between the tag and the reader, i.e., A , B , C and D . However, they are mainly used as the input of a hash function that is not possible to invert efficiently. The only way to achieve such an inversion should be to use $C = H(r_t^i \oplus r_r^i \oplus PID^i) \oplus PID^{i+2}$ to determine PID^{i+2} . However, it is randomized by $H(r_t^i \oplus r_r^i \oplus PID^i)$ for which the adversary has no knowledge and cannot control. Hence, the adversary cannot efficiently determine PID^i and PID^{i+1} .

Impersonation Attack: Given the fact that the adversary cannot determine PID^i and PID^{i+1} , to impersonate a tag to the reader, the adversary should at least generate a valid A . However, to produce it, the adversary ought to appreciate the knowledge of PID^i , since the previously eavesdropped messages cannot be used also due to the fresh r_r^i and r_t^i that are generated in each new session, the adversary cannot do this attack efficiently. A similar argument is also valid and relevant for the reader to the tag impersonation attack.

Replay Attack: Considering the fact that all messages that are critical to authenticate the tag by the reader or the reader by the tag are randomized by both parties, it is not possible to apply a replay attack against the proposed protocol.

Tag Cloning Attack: Assuming that the used PUF behaves randomly, it is impossible to clone a tag even when the adversary compromises the tag to access TID^i , TID^{i-1} and N^i . The reason for this comes from the fact that it is not possible to implement identical PUF in different tags.

Desynchronization Attack: To desynchronize the tag and the reader, the reader should be able to impersonate the reader to the tag or the tag to the reader, which is not possible. Another approach would be by blocking the last message sent from the tag to the reader. In this way, the tag has updated its records but not the reader. However, given that the tag also keeps a record of the old data; in this case, it can be synchronized based on them.

Forward/Backward Security: Given that the tag and the reader share two parameters PID^i and PID^{i+1} and since, in each round, only one of them is changed while the tag keeps a history of old parameters, we define our forward security model as the case when the adversary has all parameters of session x , misses two successful sessions and aims to recover parameters in the $(x + 2)^{th}$ session. It is clear that, in two consecutive sessions, both PID^i and PID^{i+1} are changed through PUF randomly, and, without them, the adversary has no advantage over a blind adversary. Hence, the adversary will not be able to determine PID^i and PID^{i+1} . A similar reasoning applies to the backward security of the protocol.

6.2.2. Formal Verification Using a Scyther Tool

Scyther [43] is a widely accepted tool to evaluate the security correctness of a protocol. It provides a graphical user interface to facilitate the analysis of complex attack scenarios on the target protocol. We applied the Scyther tool to verify whether our security assertions for the proposed protocol holds or not. We defined two identities over the protocol, i.e., *tag* and *reader* in Scyther. Then, we implemented each role with its corresponding task, followed by the implementations as represented in Appendix A. It is worth noting that we fixed the number of protocol runs 100 times and also fixed the search pruning to "Find all attacks" and the matching type of Scyther tool to "Find all type flaws".

We made our *Secret* claims on every message that was sent and received on both ends. According to Table 2, the Scyther tool could not find any attack within bounds, thus establishing that our security allegations were founded.

Table 2. The proposed protocol verification results using the Scyther tool.

Claim	Status	Comments		
improved R	improved,R1	Secret ID	OK	No attacks within bounds.
	improved,R2	Niagree	OK	No attacks within bounds.
	improved,R3	Nisynch	OK	No attacks within bounds.
	improved,R4	Alive	OK	No attacks within bounds.
	improved,R5	Weakagree	OK	No attacks within bounds.
T	improved,T1	Secret PIDi	OK	No attacks within bounds.
	improved,T2	Secret PIDip1	OK	No attacks within bounds.
	improved,T3	Niagree	OK	No attacks within bounds.
	improved,T4	Nisynch	OK	No attacks within bounds.
	improved,T5	Alive	OK	No attacks within bounds.
	improved,T6	Weakagree	OK	No attacks within bounds.

In terms of a tag's weight and hardware complexity of the improved protocol, the protocol could belong to lightweight protocols. Accordingly, because PUF and hash functions have been used in the design of our proposed protocol, we can control the number of gates in the range of lightweight tags. Since a tag with less than 3000 GE [44] is lightweight, by utilizing a SPONGNET hash function with around 800 GE and some bitwise functions such as XOR, the total number of gates is less than 3000 and the proposed tag can satisfy the limitation of lightweight protocols. In addition, in terms of the security level, we compared the improved protocol with some other protocols in Table 3. The results show that our protocol could address the security issues of other protocols and is resistant against IoT attacks.

Table 3. The security comparison of the improved protocol to other protocols against IoT attacks.

Protocols	Impersonation	Traceability	Disclosure	Desynchronization
Sadeghi et al. [25]	×	×	✓	✓
Aysu et al. [32]	✓	×	×	✓
Van Herrewege et al. [30]	✓	✓	×	✓
Kulseng et al. [11]	✓	✓	×	×
Xu et al. [10]	✓	✓	×	×
Improved Protocol	✓	✓	✓	✓

✓: Resistant ×: Non-resistant.

7. Conclusions

With the exponential increase of IoT connected devices, the importance of security risks is raised by the academic and professional community as any connected object becomes a computing device and a potential target for an attack. In developing strategies to identify and prevent intrusions and threats, which are more likely to increase, researchers around the world are working on securing all the layers of the IoT infrastructure. Along these trends, since the commercial introduction by Verayo in the early 2008 of the first silicon chips equipped with PUF, the market has seen some growth with other companies (e.g., Intrinsic ID, Quantum Trace, Invia) now developing in this marketplace. While there are no doubts that PUF constitutes a very promising avenue to ensure device security, caution should always be brought to constantly evolving security protocols. In line with this issue, Xu et al. proposed a lightweight RFID mutual authentication protocol based on PUF, for ensuring mutual tag-reader verification and preventing clone attacks. While the authors claimed that their protocol was efficient to protect RFID systems, we found it to still be vulnerable and presented how a desynchronization attack and several secret disclosure attacks could be performed on Xu et al.'s protocol. Finally, we discussed

why it is better to try to employ secure lightweight encryption functions in designing secure protocols rather than ad hoc designs, by proposing a secure hash based protocol.

Author Contributions: The authors contributed equally to this work.

Funding: This research received no external funding.

Acknowledgments: We would like to thank the anonymouse reviewers for their suggestions on improving the presentation of the paper and also their technical suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Implementation of the Proposed Protocol in the Scyther Tool

```

hashfunction h;
const xor:Function;
const con:Function;
const add:Function;
const puf:Function;
secret ID;
secret Ni;
secret TIDi;
secret PIDi;
secret PIDip1;
macro TIDip1=h(xor(PIDip1,TIDi));
macro A=h(con(xor(puf(xor(ID,Ni)),rri),rti));
macro B=h(con(xor(PIDip1,rti),xor(rri,TIDip1)));
macro C=xor(h(xor(rti,rri,puf(xor(ID,Ni))))),puf(xor(ID,add(Ni,2))));
macro D=h(con(xor(C, rti),xor(rri,puf(xor(ID,add(Ni,1))))));
protocol improved(R,T){
role R{
secret Ni;
secret TIDi;
secret PIDi;
secret PIDip1;
secret ID;
fresh rri;
var rti;
send_1(R,T,rri);
recv_2(T,R,A,TIDi,rti);
send_3(R,T,B);
recv_4(T,R,con(C,D));
claim(R,Secret,ID);
claim(R,Niagree);
claim(R,Nisynch);
claim(R,Alive);
claim(R,Weakagree);
}
role T{
fresh rti;
var rri;
secret Ni;
secret TIDi;
secret PIDi;

```



```

secret ID;
secret PIDip1;
recv_1(R,T,rri);
send_2(T,R,A,TIDi,rTi);
recv_3(R,T,B);
send_4(T,R,con(C,D));
claim(T,Secret,PIDi);
claim(T,Secret,PIDip1);
claim(T,Niagree);
claim(T,Nisynch);
claim(T,Alive);
claim(T,Weakagree);
}}

```

References

1. Weiser, M. Hot topics-ubiquitous computing. *Computer* **1993**, *26*, 71–72. [CrossRef]
2. Pelino, M.; Hammond, J.; Dai, C.; Miller, P.; Belissent, J.; Ask, J.; Fenwick, N.; Gillett, F.; Husson, T.; Maxim, M.; et al. Predictions 2018: IoT Moves From Experimentation To Business Scale. Available online: <https://www.forrester.com/report/Predictions+2018+IoT+Moves+From+Experimentation+To+Business+Scale/-/E-RES139752> (accessed on 12 October 2018).
3. Markit, I. Number of Connected IoT Devices Will Surge to 125 Billion by 2030. Available online: <https://technology.ihs.com/596542/number-of-connected-iot-devices-will-surge-to-125-billion-by-2030-ihs-markit-says> (accessed on 12 October 2018).
4. Hung, M. Leading to IoT: Gartner Insights on How to Lead in a Connected World. Available online: https://www.gartner.com/imagesrv/books/iot/iotEbook_digital.pdf (accessed on 12 October 2018).
5. Wang, K.H.; Chen, C.M.; Fang, W.; Wu, T.Y. A secure authentication scheme for Internet of Things. *Perv. Mob. Comput.* **2017**, *42*, 15–26. [CrossRef]
6. EPCglobal, G. *EPC Radio-Frequency Identity Protocols Generation-2 UHF RFID; Specification for RFID Air Interface Protocol for Communications at 860 MHz–960 MHz*; EPCglobal Inc.: Menlo Park, CA, USA, 2013.
7. Karygiannis, T.; Eydt, B.; Barber, G.; Bunn, L.; Phillips, T. Guidelines for securing radio frequency identification (RFID) systems. *NIST Spec. Pub.* **2007**, *80*, 1–154.
8. Gassend, B.; Clarke, D.; Van Dijk, M.; Devadas, S. Silicon physical random functions. In Proceedings of the 9th ACM Conference on Computer and Communications Security, Washington, DC, USA, 18–22 November 2002; Volume 1, pp. 148–160.
9. RFID Startup Offers Enhanced Security for Tag Chips. Available online: <https://www.rfidjournal.com/articles/view?7042> (accessed on 12 October 2018).
10. Xu, H.; Ding, J.; Li, P.; Zhu, F.; Wang, R. A Lightweight RFID mutual authentication protocol based on physical unclonable function. *Sensors* **2018**, *18*, 760. [CrossRef] [PubMed]
11. Kulseng, L.; Yu, Z.; Wei, Y.; Guan, Y. Lightweight mutual authentication and ownership transfer for RFID systems. In Proceedings of the 2010 Proceedings IEEE INFOCOM, San Diego, CA, USA, 14–19 March 2010; Volume 1, pp. 1–5.
12. Sector, S.; Itu, O. *SERIES Y: GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS Next Generation Networks—Frameworks and Functional Architecture Models*; International Telecommunication Union: Geneva, Switzerland, 2012.
13. Lu, Y.; Papagiannidis, S.; Alamanos, E. Internet of Things: A systematic review of the business literature from the user and organisational perspectives. *Technol. Forecast. Soc. Chang.* **2018**, *23*, 31–41. [CrossRef]
14. Krotov, V. The Internet of Things and new business opportunities. *Bus. Horiz.* **2017**, *60*, 831–841. [CrossRef]
15. IoT Business Models Framework UNIFY-IoT. Available online: http://www.unify-iot.eu/wp-content/uploads/2016/10/D02_01_WP02_H2020_UNIFY-IoT_Final.pdf (accessed on 12 October 2018).

16. The Internet of Things Reference Model, White paper. Available online: https://www.cisco.com/c/dam/global/en_ph/assets/ciscoconnect/pdf/bigdata/jim_green_cisco_connect.pdf (accessed on 12 October 2018).
17. Qi, J.; Yang, P.; Min, G.; Amft, O.; Dong, F.; Xu, L. Advanced internet of things for personalised healthcare systems: A survey. *Pervas. Mob. Comput.* **2017**, *41*, 132–149. [[CrossRef](#)]
18. Bendavid, Y. Positioning RFID technologies in the enterprise information systems portfolio: A case in supply chain management. *Int. J. Autom. Technol.* **2012**, *4*, 11–24.
19. Tan, H.; Choi, D.; Kim, P.; Pan, S.; Chung, I. Secure Certificateless Authentication and Road Message Dissemination Protocol in VANETs. *Wirel. Commun. Mob. Comput.* **2018**, *2018*, 7978027. [[CrossRef](#)]
20. Afifi, M.; Zhou, L.; Chakrabartty, S.; Ren, J. Dynamic authentication protocol using self-powered timers for passive Internet of Things. *IEEE Internet Things J.* **2018**, *5*, 2927–2935. [[CrossRef](#)]
21. Tan, H.; Choi, D.; Kim, P.; Pan, S.; Chung, I. Comments on “Dual Authentication and Key Management Techniques for Secure Data Transmission in Vehicular Ad Hoc Networks”. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 2149–2151. [[CrossRef](#)]
22. Shen, J.; Tan, H.; Zhang, Y.; Sun, X.; Xiang, Y. A new lightweight RFID grouping authentication protocol for multiple tags in mobile environment. *Multimed. Tools Appl.* **2017**, *76*, 22761–22783. [[CrossRef](#)]
23. Gope, P.; Lee, J.; Quek, T.Q. Lightweight and Practical Anonymous Authentication Protocol for RFID Systems Using Physically Unclonable Functions. *IEEE Trans. Inf. Forens. Secur.* **2018**, *13*, 2831–2843. [[CrossRef](#)]
24. Cao, Y.; Zhao, X.; Ye, W.; Han, Q.; Pan, X. A Compact and Low Power RO PUF with High Resilience to the EM Side-Channel Attack and the SVM Modelling Attack of Wireless Sensor Networks. *Sensors* **2018**, *18*, 322. [[CrossRef](#)] [[PubMed](#)]
25. Sadeghi, A.R.; Visconti, I.; Wachsmann, C. PUF-enhanced RFID security and privacy. *SECSI* **2010**, *110*, 146–149.
26. Kardaş, S.; Kiraz, M.S.; Bingöl, M.A.; Demirci, H. A novel RFID distance bounding protocol based on physically unclonable functions. In *International Workshop on Radio Frequency Identification: Security and Privacy Issues*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 1, pp. 78–93.
27. Akgun, M.; Caglayan, M.U. PUF based scalable private RFID authentication. In Proceedings of the 2011 Sixth International Conference on Availability, Reliability and Security (ARES), Vienna, Austria, 22–26 August 2011; Volume 1, pp. 473–478.
28. Akgün, M.; Çaglayan, M.U. Providing destructive privacy and scalability in RFID systems using PUFs. *Ad Hoc Netw.* **2015**, *32*, 32–42. [[CrossRef](#)]
29. Kardaş, S.; Çelik, S.; Yıldız, M.; Levi, A. PUF-enhanced offline RFID security and privacy. *J. Netw. Comput. Appl.* **2012**, *35*, 2059–2067. [[CrossRef](#)]
30. Van Herrewege, A.; Katzenbeisser, S.; Maes, R.; Peeters, R.; Sadeghi, A.R.; Verbauwhede, I.; Wachsmann, C. Reverse fuzzy extractors: Enabling lightweight mutual authentication for PUF-enabled RFIDs. In *International Conference on Financial Cryptography and Data Security*; Springer: Berlin/Heidelberg, Germany, 2012; Volume 1, pp. 374–389.
31. Moriyama, D.; Matsuo, S.; Yung, M. PUF-based RFID authentication secure and private under memory leakage. *IACR Cryptol. ePrint Arch.* **2013**, *3*, 61–83.
32. Aysu, A.; Gulcan, E.; Moriyama, D.; Schaumont, P.; Yung, M. End-to-end design of a PUF-based privacy preserving authentication protocol. In *International Workshop on Cryptographic Hardware and Embedded Systems*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 556–576.
33. Huth, C.; Aysu, A.; Guajardo, J.; Duplys, P.; Güneysu, T. Secure and Private, yet Lightweight, Authentication for the IoT via PUF and CBKA. In *International Conference on Information Security and Cryptology*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 28–48.
34. Ray, B.; Howdhury, M.; Abawajy, J.; Jesmin, M. Secure object tracking protocol for Networked RFID Systems. In Proceedings of the 2015 16th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Takamatsu, Japan, 1–3 June 2015; Volume 1, pp. 1–7.
35. Li, Y.; Deng, R.H.; Bertino, E. RFID security and privacy. *Synth. Lect. Inf. Secur. Privacy Trust* **2013**, *4*, 1–157. [[CrossRef](#)]

36. Beaulieu, R.; Shors, D.; Smith, J.; Treatman-Clark, S.; Weeks, B.; Wingers, L. The SIMON and SPECK lightweight block ciphers. In Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, 7–11 June 2015; pp. 175:1–175:6.
37. Beierle, C.; Jean, J.; Kölbl, S.; Leander, G.; Moradi, A.; Peyrin, T.; Sasaki, Y.; Sasdrich, P.; Sim, S.M. The SKINNY Family of Block Ciphers and Its Low-Latency Variant MANTIS. In Proceedings of the 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, 14–18 August 2016.
38. Aumasson, J.; Jovanovic, P.; Neves, S. NORX: Parallel and Scalable AEAD. In Proceedings of the 19th European Symposium on Research in Computer Security, Wroclaw, Poland, 7–11 September 2014.
39. Bogdanov, A.; Knezevic, M.; Leander, G.; Toz, D.; Varici, K.; Verbauwhede, I. SPONGENT: A Lightweight Hash Function. In *CHES*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6917, pp. 312–325.
40. Lightweight Hash Functions. Available online: https://www.cryptolux.org/index.php/Lightweight_Hash_Functions#cite_note-BKLT11-22 (accessed on 10 November 2018).
41. Guo, J.; Peyrin, T.; Poschmann, A. The PHOTON Family of Lightweight Hash Functions. In *CRYPTO*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6841, pp. 222–239.
42. Aumasson, J.; Henzen, L.; Meier, W.; Naya-Plasencia, M. Quark: A Lightweight Hash. *J. Cryptol.* **2013**, *26*, 313–339. [[CrossRef](#)]
43. Cremers, C.J.F. The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols. In *Computer Aided Verification*; Gupta, A., Malik, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 414–418.
44. Sundaresan, S.; Doss, R.; Piramuthu, S.; Zhou, W. A robust grouping proof protocol for RFID EPC C1G2 tags. *IEEE Trans. Inf. Forens. Secur.* **2014**, *9*, 961–975. [[CrossRef](#)]



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).