

On Privacy of RFID-Based Authentication Protocols

Ferucio Laurențiu Țiplea^a

Faculty of Computer Science, “Alexandru Ioan Cuza” University of Iași, Romania

Keywords: RFID Protocol, Privacy, Indistinguishability.

Abstract: RFID-based authentication protocols have started increasingly being used in various fields, such as tracking assets, managing stock, controlling quality processes, and automotive and healthcare services. In addition to authentication (unilateral or mutual), it is necessary that these protocols also ensure a certain level of privacy. Very often, privacy properties are studied ad hoc or informally. In this paper, we highlight two scenarios that, once identified in such protocols, show us that they cannot satisfy a certain level of privacy in the Hermans-Pashalidis-Vercauteren-Preneel (HPVP) privacy model. For each scenario, general results are presented and exemplified by existing protocols. We then analyze a recent authentication protocol based on simulatable PUFs and prove that a simplified variant of it achieves strong privacy in the HPVP model.

1 INTRODUCTION


We probably use radio frequency identification (RFID) technology daily in public transportation or payment in stores, most of the time without being aware. This technology uses the electromagnetic field to read information from a tag associated with the object or entity we want to identify.

Officially, the RFID technology was invented in 1983 by Charles Walton when he filed the first patent with the word “RFID”, even if it was used in various forms long before that. Near Field Communication (NFC), also based on RFID, debuted in 2002 and has since continued to develop. In the last 20 years, we have witnessed an impetuous development of RFID technology. Its applications are increasingly diverse, from big retail chains worldwide to tracking assets, managing stock, controlling quality processes, and automotive and healthcare services.

The emergence and development of the Internet of Things has led to new and multiple applications of RFID. On the other hand, we are witnessing a diversification of the use of RFID technology. We are thus talking about a backend server-based RFID architecture, where the tag reader communicates securely with a server, and a server-less RFID architecture suitable for identification and verification by tag readers that are mobile and offline. The development of cloud technology allowed the replacement of the processing provided by the backend server by processing data in

the cloud. This makes it possible to use mobile tag readers but also fixed ones. Another aspect worthy of consideration is the appearance of physically unclonable functions (PUFs), which are chips that theoretically cannot be cloned, behave like pseudo-random functions, and are tamper-resistant. They can be included in tags, offering significant advantages in the identification and authentication process.

In all this wealth of applications, the central role of RFID is identification (with the provision of information about the identified object) and authentication. To this, the RFID protocols must meet specific security and privacy requirements. Various security and privacy models have been developed for cases where RFID technology is used with a backend server securely connected to the tag reader. Vaudenay’s model (Vaudenay, 2007; Paise and Vaudenay, 2008) and the Hermans-Pashalidis-Vercauteren-Preneel (HPVP) model (Hermans et al., 2011; Hermans et al., 2014) are two of the most important. These models can be extended to other RFID architectures proposed in recent years because, implicitly, the security and privacy properties between tag and reader must be preserved somehow. The communication between the reader and the server, when the communication channel is not secure, must be further analyzed. However, if the reader-tag communication is not private, it cannot be private regardless of how we extend the architecture to capture aspects of mobility, cloud computing, etc.

^a  <https://orcid.org/0000-0001-6143-3641>

Contribution. Even though we currently have sufficiently mature security and privacy RFID models, many authors still propose authentication protocols based on RFID for which the analysis of security and privacy properties is carried out ad hoc or informally. It has often been found that such protocols have vulnerabilities, and their fix leads to other protocols with vulnerabilities. In this context, establishing templates that quickly highlight protocols that cannot be secure or private, can significantly help RFID protocol designers. However, achieving this goal may not be simple, especially for the protocols whose analysis was carried out informally. Reporting to a reputable RFID security and privacy model can solve the problem in such a situation. Thus, in this paper, we will consider the HPVP model as a reference model and establish some “templates” that, once detected in a security protocol, guarantee that the protocol cannot reach a certain level of privacy in the HPVP model. For each template, we provide theoretical proof in the HPVP model and exemplify it abundantly through RFID protocols recently proposed in the specialized literature. We then analyze a recent authentication protocol (Gao et al., 2022) based on simulatable PUFs and prove that a simplified variant of it achieves strong privacy in the HPVP model.

Paper Organization. The paper’s second section recalls a few basic definitions and notations in cryptography, while the third section deals with the RFID system concept, briefly presents the HPVP privacy model, and shows how the technology of physically unclonable functions (PUF) can be used with RFID systems. In the fourth section, we discuss the consequences on the properties of privacy of the non-randomization of the tag identifier and the “unprotected” use of global temporary variables within the RFID tags. The fifth section is dedicated to a recently proposed class of RFID authentication protocols based on simulatable PUFs. We discuss one of these protocols and propose an improvement, together with the complete proof of strong privacy in the HPVP model. In the last section, we present the conclusions of the paper.

2 SOME BASIC DEFINITIONS AND NOTATION

We will briefly mention some standard cryptography concepts and notations (for details, the reader is directed to (Katz and Lindell, 2020)).

We use in our exposition *probabilistic polynomial time* (PPT) algorithms \mathcal{A} as defined in (Sipser,

2012) that can consult *oracles*. For a set A , $a \leftarrow A$ means that a is uniformly at random chosen from A . If \mathcal{A} is a probabilistic algorithm, then $a \leftarrow \mathcal{A}$ means that a is an output of \mathcal{A} for some given input. The asymptotic approach to security makes use of security parameters, denoted by λ in our paper. A positive function $f(\lambda)$ is called *negligible* if for any positive polynomial $poly(\lambda)$ there exists n_0 such that $f(\lambda) < 1/poly(\lambda)$, for any $\lambda \geq n_0$. $f(\lambda)$ is called *overwhelming* if $1 - f(\lambda)$ is negligible.

Assume that \mathcal{K} is a key space. A family of functions indexed by \mathcal{K} is a function F that associates to any key $K \in \mathcal{K}$ a function $F_K : \{0, 1\}^{\ell_1(|K|)} \rightarrow \{0, 1\}^{\ell_2(|K|)}$, where $|K|$ is the size of K and ℓ_1 and ℓ_2 are two polynomials given for F . We usually denote F by $F = (F_K)_{K \in \mathcal{K}}$. We say that F is a *pseudo-random function* (PRF) if it is efficiently computable and the input-output behavior of any function chosen at random from F is computationally indistinguishable from that of a random function. To prove that F is a PRF, we usually assume the existence of a *challenger* \mathcal{C} that plays the following security game parameterized by a *security parameter* λ with an adversary \mathcal{A} :

1. \mathcal{C} randomly chooses $b \leftarrow \{0, 1\}$;
2. if $b = 1$ then \mathcal{C} randomly chooses a key K of size λ from the key space and sets $f = F_K$; otherwise, \mathcal{C} randomly chooses f from the space of all functions from $\{0, 1\}^{\ell_1(\lambda)}$ to $\{0, 1\}^{\ell_2(\lambda)}$;
3. \mathcal{C} provides oracle access to f for \mathcal{A} ;
4. At some point, \mathcal{A} outputs a bit b' .

The adversary \mathcal{A} wins the game if $b' = b$.

Now, F is a PRF if it is efficiently computable and the probability to win the above security game is negligible close to $1/2$, for all adversaries.

3 PRIVACY OF RFID-BASED SYSTEMS

This section introduces the basic concepts we will use further, such as the RFID system, physically unclonable functions, and privacy properties.

3.1 RFID System

For the RFID system concept, we will especially follow (Vaudenay, 2007; Hermans et al., 2011; Hermans et al., 2014). Informally, an RFID system consists of a *reader*, a set of *tags*, and a radio frequency *communication protocol* between reader and tags. The reader, which securely communicates with a database where

information on the tags in the system is stored, has no restrictions from a computational point of view. The tags, however, are devices with very low storage capacity and computing power, generally supplied with energy by the reader when it is in their vicinity. The tag memory is divided into:

1. *Permanent memory*, which stores the state values of the tag;
2. *Temporary or volatile memory*, which can be viewed as a set of *temporary (volatile) variables* used to carry out the calculations required by the communication protocol. There are two types of temporary variables:
 - (a) *Local temporary variables*, used by tags only to do computations in a given protocol step;
 - (b) *Global temporary variables*, that get values in a given protocol step and are used in another protocol step.

Let \mathcal{R} be a *reader identifier* and \mathcal{T} be a set of *tag identifiers* whose cardinal is polynomial in some security parameter λ . An *RFID scheme over* $(\mathcal{R}, \mathcal{T})$ (Vaudenay, 2007) is a triple $\Sigma = (\text{SetupR}, \text{SetupT}, \text{Ident})$ of PPT algorithms, where:

1. $\text{SetupR}(\lambda)$ sets the reader. Starting with a security parameter λ , it generates a triple (pk, sk, DB) consisting of a key pair (pk, sk) and an empty database DB . pk is public, while sk is kept secret by reader;
2. $\text{SetupT}(pk, ID)$ initializes the tag identified by ID . It outputs an initial tag state S and a tag-specific secret K . The pair (ID, K) is stored in the reader's database;
3. $\text{Ident}(pk; \mathcal{R}(sk, DB); ID(S))$ is a communication protocol between the reader identified by \mathcal{R} (with its private key sk and database DB) and a tag identified by ID (with its state S) in which the reader ends with an output consisting of ID or \perp . The tag may end with no output (*unilateral authentication*), or it may end with an output consisting of OK or \perp (*mutual authentication*).

The tag's state S may be a vector of elements, and the same K . Moreover, S and K may not necessarily be disjoint. Their common part, if any, will be called the *DB-state* of the tag; it will be denoted by $S|_{DB}$. Among other things, the *DB-state* is used for synchronization between reader and tag.

The *correctness* of an RFID scheme refers to the honest behavior of the reader and tag in a complete protocol session. More precisely, regardless of how the system is set up, after each complete and honest execution of the interactive protocol one of the two cases holds with overwhelming probability:

- If the tag is legitimate, the reader outputs tag's identity (and the tag outputs OK , in case of mutual authentication);
- If the tag is illegitimate, the reader outputs \perp .

The communication protocol is an alternating sequence of reader-to-tag and tag-to-reader communication steps in which the first step can be taken by either of them. When the reader sends a message m to the tag, we will often say that the reader queries the tag on m . When the first protocol step is taken by the tag, we will say that the tag answers to the empty query (this corresponds to the tag being powered by the reader).

An *RFID system* is an instantiation of an RFID scheme.

3.2 The HPVP Privacy Model

Considerable effort has been put into the development of security and privacy models for RFID systems. Vaudenay's model (Vaudenay, 2007) and the Hermans-Pashalidis-Vercauteren-Preneel (HPVP) model (Hermans et al., 2011; Hermans et al., 2014) are two of them, with major impact in the study of security and privacy properties of RFID systems. The HPVP model borrows the adversary model from Vaudenay's model, keeps the same approach to the security property, but treats privacy in a different way. If in Vaudenay's model privacy is based on indistinguishability between the RFID system instrumented by a challenger and the RFID system instrumented by a blinder (who does not know the secret elements in the system), the HPVP model treats privacy through indistinguishability between tags in the RFID system instrumented by a challenger. This second approach is closer to the security approach of the usual encryption systems.

The adversary in the HPVP model is allowed to query the following oracles:

- $\text{CreateReader}()$: Creates a new reader, and a unique reference R to it is returned;
- $\text{CreateTag}(ID)$: Creates a tag with the identifier ID by calling $\text{SetupT}(pk, ID)$. The tag is registered in the server's database (that is, the oracle creates only legitimate tags). Moreover, duplicate tags with the same ID are accepted. A unique reference T to the tag is returned;
- $\text{RegisterTag}(T, R)$: Registers the tag T with the reader R ;
- $\text{Launch}(R)$: Launches a new protocol instance with the reader R , assigns a unique identifier π to it, and outputs π ;

- $DrawTag(T_0, T_1)$: Generate a fresh virtual tag reference $vtag$ that refers to either the left tag T_0 or to the right tag T_1 , depending on the privacy game where the oracle is queried. The triple $(vtag, T_0, T_1)$ is included in a list Γ of drawn tags, and $vtag$ is returned by the oracle.

The oracle returns \perp in any of the following cases:

- If one of the two tags is in the insider list;
- If one of the two tags is registered with a different set of readers than the other tag;
- If T_0 (T_1) is already referenced as the left-hand (right-hand) side tag in Γ ;
- $Free(vtag)$: This oracle resets (erases the temporary state of) the tag referenced by $vtag$ and removes the corresponding triple from Γ ;
- $SendTag(m, vtag)$: Outputs the tag's answer when the message m is sent to the tag referred to by $vtag$. When m is the empty message, this oracle outputs the first message of the protocol instance π , assuming that the tag does the first step in the protocol;
- $SendReader(R, m, \pi)$: Outputs the R 's reader answer when the message m is sent to it as part of the protocol instance π . When m is the empty message, abusively but suggestively denoted by \emptyset , this oracle outputs the first message of the protocol instance π , assuming that the reader does the first step in the protocol;
- $Result(\pi)$: Outputs \perp if in session π the reader has not yet made a decision on tag authentication (this also includes the case when the session π does not exist), 1 if in session π the reader authenticated the tag, and 0 otherwise (this oracle is both for unilateral and mutual authentication);
- $Corrupt(T)$: Outputs the full (permanent and temporary) state of the tag T . Remark that the corruption is with respect to a tag, not a virtual tag. Otherwise, it would be easy for an adversary with corruption capabilities to distinguish between the left and the right privacy games (Hermans et al., 2011; Hermans et al., 2014);
- $CreateInsider(ID)$: Creates a tag and returns a unique reference T to it and its full state. The tag is included in a list of insider tags. This oracle is a method of giving the adversary the internal state of a tag but without considering that the adversary has used the corruption oracle.

We will now classify the adversaries as follows:

1. Adversaries with no access to $CreateInsider$. These are further classified according to the way the $Corrupt$ oracle is used:

- (a) *Weak adversaries*: no access to the $Corrupt$ oracle;
- (b) *Forward adversaries*: once they access the $Corrupt$ oracle, the only oracle they can access is $Corrupt$;
- (c) *Destructive adversaries*: after accessing the $Corrupt(T)$ oracle, the tag T is destroyed (but the information about T is kept in the database);
- (d) *Strong adversaries*: no restrictions;
- (e) *Narrow weak (narrow forward, narrow destructive, narrow strong)*: as above but the adversary has no access to the $Result$ oracle;

2. Adversaries with access to $CreateInsider$. The power of a destructive or strong adversary does not increase if he is given access to the $CreateInsider$ oracle. As a result, we distinguish the following two classes of adversaries:

- (a) *Weak-insider adversaries*: weak adversaries with supplementary access to the $CreateInsider$ oracle;
- (b) *Forward-insider adversaries*: forward adversaries with supplementary access to the $CreateInsider$ oracle.

Some authors (Hermans et al., 2014) refer to the classes of adversaries from (1)(a)-(d) as being *wide* in the sense that the adversaries in these classes may consult the $Result$ oracle.

The practical use of RFID schemes requires that they provide some level of security and privacy. Our paper focuses only on the privacy property. However, for completeness, we briefly recall the security property, too. Security means that no strong adversary has more than a negligible probability to make the reader authenticate an uncorrupted legitimate tag without having any tag authentication matching conversation. When the RFID scheme is with mutual authentication, besides the above requirement, it is asked that no strong adversary has more than a negligible probability to make an uncorrupted legitimate tag to authenticate the reader without having any reader authentication matching conversation.

As with respect to privacy, define the experiment $RFID_{\mathcal{A}, \Sigma}^{hpvp-prv-b}(\lambda)$, where $b \in \{0, 1\}$, \mathcal{A} is an adversary, and Σ is the RFID scheme, as follows:

$$RFID_{\mathcal{A}, \Sigma}^{hpvp-prv-b}(\lambda)$$

1. Set up reader;
2. \mathcal{A} gets pk ;
3. \mathcal{A} is given access to oracles. For $b = 0$, all temporary identities refer to the left tag drawn by adversary, while for $b = 1$ they refer to the right tag;

4. \mathcal{A} outputs a bit b' ;
5. Return 1 if $b' = b$, and 0, otherwise.

For $b = 0$ ($b = 1$) this experiment will also be called the *left (right) privacy game*.

The advantage of \mathcal{A} against Σ is defined as

$$Adv_{\mathcal{A}, \Sigma}^{hpvp-prv}(\lambda) = |P(RFID_{\mathcal{A}, \Sigma}^{hpvp-prv-0}(\lambda) = 1) + P(RFID_{\mathcal{A}, \Sigma}^{hpvp-prv-1}(\lambda) = 1) - 1|$$

Now, we say that Σ is *C-private*, where C is a class of adversaries, if the advantage of any adversary in class C against Σ is negligible.

The previously defined adversary classes lead to a ranking of the privacy property of RFID schemes as shown in the diagram in Figure 1.

3.3 PUF-Based RFID Systems

One of the primary reasons that led to the development of *physically unclonable functions* (PUFs) was to find a method of protecting the secret keys against software and physical attacks (Rührmair and van Dijk, 2013). A PUF can be considered a disordered physical system that can be challenged with external stimuli (challenges) to which it will react with corresponding responses. Unlike standard digital systems, the reaction of a PUF to a challenge depends on the micro- or nanoscale structural disorder of the PUF. Ideally, it is assumed that:

1. This disorder cannot be cloned or reproduced precisely, even by PUF's original manufacturer;
2. The response r of the PUF to a challenge c is uniquely and uniformly at random chosen from the space of possible responses;
3. PUFs are tamper-evident (fully invasive attacks either damage or alter the functional behavior).

As a result, an ideal PUF defines a unique function P . The pairs (c, r) with $P(c) = r$ are called *challenge-response pairs* (CRP).

The (ideal) properties of PUFs mentioned above, as well as the technological progress aimed at achieving these properties, led to the proposal of security protocols that include them in various forms. Thus, we can mention protocols for oblivious transfer, bit commitment, key exchange, key generation, or authentication (Rührmair and van Dijk, 2013; Delvaux, 2017; Gope and Sikdar, 2021). For example, (Delvaux, 2017) reviews key generators and authentication protocols based on PUFs proposed up to 2016. Among newer protocols, we mention (Țiplea and Hristea, 2021; Gao et al., 2022).

As far as we know, the first use of PUFs in RFID systems appears in (Sadeghi et al., 2010b; Sadeghi et al., 2010a) to provide a solution to the problem of finding a private destructive RFID system in Vaudey's model. Later, the use of PUFs in RFID systems gained momentum (see (Țiplea et al., 2021) for an ample discussion on this topic). The method of use is as follows:

- Tags are endowed with PUFs and store secret information (usually a secret key);
- When the tag authenticates itself to the reader, it interrogates the PUF and extracts that secret information, which will then be used in preparing the message for the reader.

Tags with PUFs embedded into them are usually called *PUF tags*. A *PUF-based RFID scheme* is an RFID scheme with PUF tags.

The previously discussed adversary model is trivially extended to the case of PUF tags. We only need to discuss the *Corrupt* and *CreateInsider* oracles:

1. Due to PUF's tamper-evident property, no adversary with the possibility of corrupting PUF tags can obtain the secret information stored in the PUF. So, the *Corrupt*(T) oracle returns only the full state of the tag as in the standard case;
2. Due to the non-clonability of PUFs, *CreateInsider*(ID) creates a tag with the identity ID and lets the adversary simulate its PUF through a list of randomly generated pairs. This makes this oracle have a behavior similar to that of the original approach.

As a result of these, the classification and ranking of the privacy properties in Figure 1 remains the same for the case of PUF-based RFID schemes.

4 TWO DESIGN SCENARIOS IN THE HPVP MODEL

In the last fifteen years, RFID-based authentication has diversified both as a technology and in applications. If we were talking about backend server-based RFID authentication at the beginning, we are also talking about cloud-based RFID authentication. RFID authentication applications have penetrated healthcare, IoT, and automotive. PUF technology has begun to be incorporated into RFID-based authentication techniques. Regardless of how RFID-based authentication is done or its applications, the privacy component has a dominant importance. The privacy models developed for standard RFID schemes can easily be extended to either backend server-based,

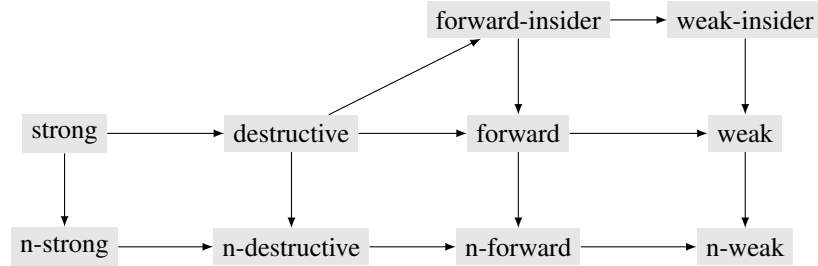


Figure 1: Privacy levels in the HPVP model: “n-p” means “narrow p” and an arrow means “implication”.

cloud-based, or PUF-based authentication. In addition, we consider that the approach to privacy properties through the HPVP model offers at least two significant advantages:

1. A rich palette of privacy properties covering the most practical applications;
2. Precise proof techniques based on indistinguishability in the same line as those used in the study of the security of other cryptographic primitives.

In this section, we will present two scenarios that, once identified in the structure of an RFID-based authentication scheme, clearly show that the scheme is not private in the HPVP model. Each scenario is rigorously argued and exemplified by RFID schemes proposed in the specialized literature.

4.1 Tag Identifiers

In most cases, the first step of an RFID protocol is performed by the reader, who sends the tag a message that can include a fresh random. The tag’s answer may or may not depend on the reader’s initial query, but what is most important is that it must include a specific information to help the reader in the tag identification process. We call this particular information a *tag identifier*.

In general, any information from the permanent memory of the tag, such as tag’s identity ID or any secret K shared by the tag and the reader, can play the role of tag identifier. Also, any message obtained by applying a function that the reader can calculate on some tag identifiers also becomes a tag identifier. For example, $h(ID)$, $h(ID, r_1)$ or $\{ID, r_1\}_K$ are tag identifiers, assuming that h is a known hash function, $\{\cdot\}_K$ denotes encryption by K , and r_1 is a random number previously transmitted by the reader and known to the tag. However, $h(ID, r_1, r_2)$ can no longer be considered a tag identifier if r_2 is a random number generated by the tag and the reader has no way to learn it. But if the reader can learn r_2 , then $h(ID, r_1, r_2)$ becomes a tag identifier.

Let’s now assume that we have an RFID protocol with at least two steps, like this:

1. In the first step of the protocol, the reader generates a random number r_1 and sends it to the tag;
2. In the second step, the tag answers with $h(ID, r_1)$.

If an adversary can interact with the tag (in the sense that he can impersonate the reader, for example), then he can query the tag with a random parameter r_1 chosen at his will, and after receiving the tag’s answer, abort the protocol. He then repeats this procedure, with the same parameter r_1 , as many times as he wants. In this way, the adversary manages to trace the tag. But if the tag had included a parameter r_2 randomly generated for each query, i.e. its answer would have been $h(ID, r_1, r_2)$, then the adversary would not have succeeded in tracing the tag because $h(ID, r_1, r_2)$ would have changed from session to session (r_2 is randomly generated each time).

We thus arrive at the following definition. We say that a *tag identifier is non-randomized* if querying the same tag with the same message two or more times in a row in order to obtain its tag identifier, the tag identifier is the same.

The non-randomization of the tag identifier leads to the total lack of privacy in the HPVP model, as shown below. To facilitate the discussion, we will assume that the tag identifier is sent to the reader in the tag’s first message (obviously, the debate can be extended).

Theorem 4.1. *Let Σ be a (PUF-based) RFID scheme for which it can be decided in deterministic polynomial time whether two tag’s answers include or not the same tag identifier. If Σ has non-randomized tag identifiers, then it does not achieve narrow weak privacy in the HPVP model.*

Proof. Assume Σ is a (PUF-based) RFID scheme as in the theorem and the reader takes the first step in the communication protocol. Consider the following narrow weak adversary \mathcal{A} that plays the following privacy game with Σ :

1. \mathcal{A} creates three tags T_1 , T_2 , and T_3 ;
2. \mathcal{A} draws (T_1, T_2) with some temporary identity $vtag_1$;

3. \mathcal{A} plays with $vtag_1$ until it gets the first message that includes the tag identifier;
4. \mathcal{A} frees $vtag_1$;
5. \mathcal{A} draws (T_1, T_3) with some temporary identity $vtag_2$;
6. \mathcal{A} plays with $vtag_2$ until it gets the first message that includes the tag identifier;
7. \mathcal{A} outputs 0 (the left privacy game) if the two messages from $vtag_1$ and $vtag_2$ include the same tag identifier, and 1, otherwise.

Step 7 above can be performed in deterministic polynomial time according to the theorem's hypothesis.

One can easily see that this adversary guesses with overwhelming probability which privacy game is playing. So, Σ does not achieve narrow weak privacy in the HPVP model. \square

There are quite a few protocols, based on PUF or not or that use cloud support, that satisfy the requirements of the previous theorem. A selection of some of these more recent ones is presented below:

1. (Xiao et al., 2016): The parameter $M1$ from the first message of the tag to the reader includes the identifier of the tag (Tid) and is non-randomized;
2. (Xu et al., 2018): The first message of the tag contains in the clear a tag identifier, namely FID , which changes much later;
3. (Fan et al., 2018): The TID identifier is sent non-randomized by tag in its first message;
4. (Fan et al., 2019): The SID identifier in M_{T1} is sent non-randomized by tag in its first message;
5. (Fan et al., 2020): The timestamp T_i identifies the tag and is sent non-randomized in $M(T_i)$ in the tag's first message (the timestamp updates later independently of the system's local time);
6. (Zhu et al., 2020): The tag sends in its first step, in the clear, the identifier SID ;
7. (Xiao et al., 2020): The tag sends in its first step, in the clear, the identifier FID^{new} ;
8. (Safkhani et al., 2020): The tag sends in its first step, in the clear, the identifier IDS_{ii} ;
9. (Kumar et al., 2023): The tag sends in its first step, in the clear, the identifier IDS .

4.2 Temporary Variables

The use of temporary variables in the construction of protocols is generally necessary. In particular, global temporary variables have a vital role because they store values calculated at a specific step in the protocol to be checked in a later step. This is the case,

for example, of challenge-and-response authentication protocols.

By corrupting devices (such as RFID tags) that use temporary variables in executing a specific protocol, the values of these variables can be obtained (at the time of corruption). This can lead to significant vulnerabilities in the protocol used. A general result on this aspect has been mentioned since 2010 (Armknrecht et al., 2010) within the RFID protocols. This result shows that it is impossible to obtain both mutual authentication and narrow forward privacy in Vaudenay's model if the adversary can obtain the global variables' value used in the authentication of the reader by the tag. This result draws our attention to the use of temporary variables (especially global ones) in identification and authentication protocols, which must be done with great care. (Tiplea and Hristea, 2021) proposes a general technique to protect global temporary variables. Another technique, with a narrow spectrum, is that in (Tiplea et al., 2022).

The HPVP privacy model considers the possibility of corruption by revealing the value of temporary variables (see the *Corrupt* oracle in Section 3). However, there is a very subtle aspect here. Namely, the *Free* oracle resets the tag referred to by the virtual tag, and the adversary cannot corrupt the virtual tag. As a result, the adversary might not obtain the values of the temporary variables by corrupting the tag. Obviously, this does not exclude the possibility that the adversary can identify the tag referred to by the virtual tag. The following general result applies exactly to this situation.

Theorem 4.2. *Let Σ be a (PUF-based) RFID scheme such that given two tags, given the sequence of messages exchanged by one of them with the reader in a protocol session until the tag has to authenticate the reader, and given the tags' full states, one can decide with overwhelming probability which of the two tags authenticates the reader. Then, Σ cannot achieve both reader authentication and narrow forward privacy in the HPVP model.*

Proof. Under the hypothesis of Theorem 4.2, we define the following narrow forward adversary \mathcal{A} :

1. \mathcal{A} creates two tags T_1 and T_2 ;
2. \mathcal{A} draws the tags under the temporary identity $vtag$;
3. \mathcal{A} simulates the protocol between the reader and $vtag$ until the last protocol step played by reader (including this step) is reached;
4. \mathcal{A} frees $vtag$;

5. \mathcal{A} corrupts both tags T_1 and T_2 and gets their full states;
6. With the reader's answer (previously obtained) and the tags full state's, \mathcal{A} will deduce which of the two tags authenticates the reader;
7. If \mathcal{A} decides that the reader is authenticated by T_1 , then it outputs 0; otherwise, it outputs 1.

One can easily see now that \mathcal{A} can distinguish between the left and right privacy games with overwhelming probability. Thus, Σ cannot achieve both reader authentication and narrow forward privacy in the HPVP model. \square

We note that the hypothesis of Theorem 4.2 (as well as its proof) do not explicitly call for the use of temporary variables, they being included in the tag's full state. So, the result from Theorem 4.2 remains valid even if (global or not) temporary variables are not used.

There are quite a few protocols that do not take global temporary variables into account in their analysis. We present some of them below:

1. (Jin et al., 2015): The protocol uses elliptic curve cryptography. The verification that the tag does in the last step to authenticate the reader can also be done by the adversary if it corrupts the tag and uses the messages sent between the tag and the reader until then. More precisely, the adversary can also calculate $e_1 = H_2(ID_{T_i}, r, C, z)$ and then check the congruence $e_1 \equiv e_1 P_R + z \pmod{n}$. As a result, the hypothesis of Theorem 4.2 holds for this protocol;
2. The following protocols mentioned in the previous section also satisfy the requirements of Theorem 4.2 (we omit the details in this regard because, anyway, they do not offer any privacy according to the previous section): (Xiao et al., 2016; Fan et al., 2018; Fan et al., 2019; Fan et al., 2020; Zhu et al., 2020; Xiao et al., 2020; Kumar et al., 2023);
3. (Aghili et al., 2019) proposed SecLAP to improve the security drawbacks of its predecessor protocol in (Fan et al., 2018). Unfortunately, the scheme uses temporary variables to authenticate the reader. Later, (Safkhani et al., 2020) shows that SecLAP is traceable and proposes an improved protocol. However, this new protocol uses the temporary variable N_{1r} (assigned in the tag's first step) to authenticate the reader (in the last step). This variable is sent in clear to the reader. So, the adversary can get it and, by corrupting the tag, can compute $IS_{ii}^{new} \parallel K_{ii}^{new}$. It is clear then that the hypothesis of Theorem 4.2 holds for this protocol;
4. (Azad and Ray, 2019): The tag assigns the variable R_2 in its first step and uses it in the last step to authenticate the reader. The adversary gets PSK_i by corrupting the tag, computes $h(PSK_i)$, gets then R_1 from M_1 , and R_2 from M_2 . It is clear then that the hypothesis of Theorem 4.2 holds for this protocol.
A particular aspect of this protocol is that the reader sends, in its first message, personalized information to the tag as if it knows to which tag it addresses. This means that the reader tries to identify the tag by opening, in the worst case, n sessions with the same tag, where n is the number of tags in the system;
5. (Adeli et al., 2023): The tag assigns $(R_t)_R$ in its first step, sends it in clear to the reader, and uses it in the last step to authenticate the reader and the cloud. By corruption, the adversary gets all the necessary information to recompute M_c . So, the hypothesis of Theorem 4.2 holds for this protocol;
6. (Wang et al., 2023): Proposes an RFID mutual authentication scheme based on matrix encryption. The scheme uses the temporary variable N_t to do mutual authentication. However, N_t is encrypted and sent to the reader. By corruption, the adversary can get the encryption/decryption keys and so it has access to N_t . It is clear then that the hypothesis of Theorem 4.2 holds for this protocol.

5 USING SIMULATABLE PUFs

In Section 3.3 we introduced the concept of ideal PUF and used it to extend the HPVP model with PUF tags. All the protocols mentioned in Section 4 use such PUFs. However, there are also protocols in which the PUF is considered "more realistically" as a non-deterministic function whose response depends on process variations, noise, environmental variables, and aging. However, it is assumed that all relevant environmental parameters are bounded, and the evaluation time of any given PUF has an upper bound. Therefore, two random evaluations of the PUF response given the same challenge might slightly vary with a Hamming distance between them bounded from above by a constant threshold. In such a case, one critical attribute of a PUF is the *reliability* of its responses, which estimates how consistently the responses can be generated against varying operating conditions.

In this section we will consider the PUF as a non-deterministic function, as described above.

A *simulatable PUF* (Rührmair and van Dijk, 2013; Gao et al., 2022) is a pair consisting of a PUF and a parameterized model *SimPUF* capable of computing a response r and its corresponding reliability confidence $conf$ in polynomial time for any given challenge c (i.e., $(r, conf) \leftarrow SimPUF(c)$), such that:

1. *SimPUF* is constructed using one-time privileged access by an authorized party in a secure environment and subsequent acquisition of *SimPUF* by any party is disabled;
2. If $r' \leftarrow PUF(c)$, then r' is indistinguishable from r in the sense that $P(r = r')$ is ϵ -close to 1;
3. The estimated $conf$ is ϵ -close to the reliability confidence of r' .

The use of a non-deterministic PUF P on a tag raises the problem of selecting the answer to a challenge c . However, having stored on the reader a *SimPUF* P' associated with P , there are procedures that can decide in polynomial time the answer of P on c . Such a procedure is *TREVERSE* proposed in (Gao et al., 2022). This uses P' for possible responses of P to c , and the correct selection is made based on a pseudo-random function F . The authentication protocol is the one in Figure 2. The server initiates the protocol by sending a challenge c to the tag. The tag queries its (non-deterministic) PUF, obtains $r \leftarrow P(c)$, and responds with $(x_1, y = F_r(x_1))$, where x_1 is a random value and F is a pseudo-random function shared by the server and tag. When the server receives the tag response, it uses P' , the model of the PUF P , and the *TRESERVE* function to determine r' , the possible response of P . The check is done by “ $y = F_{r'}(x_1)$ ”. If such a value is found, the tag is authenticated and announced. Otherwise, the protocol is aborted. In the case of tag authentication, it sends the server a random value x_2 and receives $z = F_{r'}(x_2)$. The value z is checked against $F_r(x_2)$. If the values match, the tag authenticates the server.

The protocol uses r and x_2 as temporary variables. However, the *Free* oracle resets the virtual tag, so the adversary cannot recover their values by corruption. This protocol would not be narrow forward private in Vaudenay’s model (see Theorem 3.1 in (Tiplea, 2022)). In the HPVP model, things are different with this protocol. We will show that it is strongly private in this model. Before this, we will simplify it to the protocol in Figure 3.

In the theorem below, we will assume that the PUF P is random. We draw attention to the fact that in this section, we consider the PUF as a non-deterministic process. As a result, it can answer dis-

tinct queries with the same challenge differently, but its answer conforms to the definition adopted for simulatable PUFs.

Theorem 5.1. *The mutual authentication scheme in Figure 3 provides strong privacy in the HPVP model, provided that P behaves randomly and F is a PRF.*

Proof. Let Σ be the scheme in Figure 3. Assume that Σ is not strong private in the HPVP model, and let \mathcal{A} be a strong adversary that can break Σ ’s privacy. We will show that there is an adversary \mathcal{B} that has a non-negligible advantage in the pseudo-randomness game with F . Let \mathcal{C} be a challenger for the pseudo-randomness game with F .

\mathcal{B} will simulate Σ (will be the challenger) in the privacy game that \mathcal{A} plays with Σ . So, \mathcal{B} will have to simulate the oracles for \mathcal{A} . \mathcal{B} does not know the secret parameters of the scheme but will want the simulation it performs to be indistinguishable from the real privacy game between \mathcal{A} and Σ . We will show below how the oracles are simulated:

1. \mathcal{B} keeps a list \mathcal{R} of readers that will be created by adversary, and a list of tags \mathcal{T}_R registered with each reader $R \in \mathcal{R}$. Initially, these lists are empty;
2. \mathcal{B} keeps a list \mathcal{T} of tags that will be created in the system in the order in which they are created. Each tag receives a fresh reference. We recall that the HPVP model allows the creation of several tags with the same identity. The corrupted (insider) tags will be stored in a separate list $c\mathcal{T}$ ($i\mathcal{T}$), initially empty;
3. \mathcal{B} will simulate the tag T ’s PUF as a list of challenge-response pairs. Initially, this list, denoted $P(T)$, is empty. When evaluating the PUF on c , \mathcal{B} looks in $P(T)$ a pair (c, r) , for some r . If such a pair is found, r will be returned as the value of P on c ; otherwise, a random value r is generated and (c, r) will be included in $P(T)$, returning r at the same time.

It is clear that \mathcal{B} simulates P deterministically. However, the adversary cannot distinguish between the real simulatable PUF and the one simulated by \mathcal{B} due to the properties that a simulatable PUF enjoys and the fact that the response given by the tag is randomized each time by x ;

4. \mathcal{B} keeps a list (table) Γ of active triples $(vtag, T_0, T_1)$ as specified in the oracle *DrawnTag*. The oracle *Free*($vtag$) removes $(vtag, T_0, T_1)$ from Γ . We draw attention to the fact that Γ can contain at most one triple with $vtag$ in the first position;
5. \mathcal{B} will keep a list Q of $(query, ext_answer)$ pairs, where $query$ is a query of \mathcal{A} and ext_answer is a

	Server (Reader) (SimPUF P' , PRF F)	Prover (Tag) (PUF P , PRF F)
1	$c \leftarrow \{0, 1\}^\ell$	\xrightarrow{c}
2		$r \leftarrow P(c)$ $x_1 \leftarrow \{0, 1\}^m$ $y := F_r(x_1)$ $\xleftarrow{x_1, y}$
3	If $fail \leftarrow TREVERSE(c, P', x_1, y)$ then <i>abort</i> else let r' be its output (i.e., $y = F_{r'}(x_1)$) <i>authenticate tag</i> \xrightarrow{auth}	
4		$\xleftarrow{x_2}$ $x_2 \leftarrow \{0, 1\}^m$
5	$z := F_{r'}(x_2)$	\xrightarrow{z}
6		if $z = F_r(x_2)$ then <i>authenticate server</i> else <i>abort</i>

 Figure 2: *SimPUF*-Based authentication scheme in (Gao et al., 2022).

	Server (Reader) (SimPUF P' , PRF F)	Prover (Tag) (PUF P , PRF F)
1	$c \leftarrow \{0, 1\}^\ell$	\xrightarrow{c}
2		$r := P(c)$ $x \leftarrow \{0, 1\}^m$ $y := F_r(x, 0)$ $\xleftarrow{x, y}$
3	If $fail \leftarrow TREVERSE(c, P', x, y)$ then <i>abort</i> else let r' be its output <i>authenticate tag</i> $z := F_{r'}(x, 1)$ \xrightarrow{z}	
4		if $z = F_r(x, 1)$ then <i>authenticate server</i> else <i>abort</i>

 Figure 3: *SimPUF*-Based strong private authentication scheme in the HPVP model.

possibly detailed information from which the answer to the query is extracted;

6. *CreateReader*(\cdot): \mathcal{B} generates a unique reader reference R , answers to \mathcal{A} with R , and includes R in \mathcal{R} . Moreover,

$$(\text{CreateReader}(\cdot), R)$$

is included in Q ;

7. *CreateTag*(ID): \mathcal{B} generates a fresh tag reference T , associates it with ID , initializes $P(T)$ by the empty list, includes the pair (T, ID) in \mathcal{T} , and answers to \mathcal{A} with T . Moreover,

$$(\text{CreateTag}(ID), T)$$

is included in Q ;

8. *RegisterTag*(T, R): \mathcal{B} includes T in the list \mathcal{T}_R and $(\text{RegisterTag}(T, R), \emptyset)$

in Q ;

9. *Launch*(R): \mathcal{B} generates a fresh session identifier π , returns it to \mathcal{A} , and includes

$$(\text{Launch}(R), (R, \pi))$$

in the list Q ;

10. *DrawTag*(T_0, T_1): \mathcal{B} checks if the constraints of the *DrawTag* oracle are satisfied. If they are not satisfied, the answer \perp is returned. Otherwise, \mathcal{B} generates a fresh virtual tag reference $vtag$, includes $(vtag, T_0, T_1)$ in Γ , and answers to \mathcal{A} with $vtag$. In Q the following pair is included

$$(\text{DrawTag}(T_0, T_1), \perp/vtag),$$

where $\perp/vatg$ is for the first/second case, resp.;

11. *Free(vtag)*: the triple whose first component is *vtag* is removed from Γ (if it is in Γ). In this case, the pair

$$(Free(vtag), \emptyset)$$

is included in Q ;

12. *SendTag(c, vtag)*: \mathcal{B} extracts from Γ the triple whose first component is *vtag*. If no such triple exists, the answer is \perp . Otherwise, let $(vtag, T_0, T_1)$ be this triple. \mathcal{B} searches each list $P(T_0)$ and $P(T_1)$ for a pair with c in the first position. If one of the lists does not contain such a pair, \mathcal{B} generates a random r and includes (c, r) in that list. Now suppose that $(c, r_0) \in P(T_0)$ and $(c, r_1) \in P(T_1)$. \mathcal{B} randomly generates x , queries C with $((r_0, x, 0), (r_1, x, 0))$ and returns (x, y) to \mathcal{A} . In Q is included

$$(SendTag(c, vtag), \perp/(c, r_0, r_1, x, y)),$$

depending on one of the two cases above;

13. *SendReader(R, (x, y), π)*: Since x is generated randomly at each query of a tag, and y is calculated from x through a PRF function, x and y can be found in at most one tuple (c, r_0, r_1, x, y) previously computed by \mathcal{B} when R queried some tag by c . In addition, x and y can only appear independently with negligible probability. As a result, if \mathcal{B} does not find in Q a tuple like the one above, it responds with \perp . Otherwise, it extracts the only tuple (c, r_0, r_1, x, y) , queries C with $((r_0, x, 1), (r_1, x, 1))$, and returns the answer z of C .

In Q , the pair below is included depending on one of the two cases above

$$(SendReader(R, (x, y), \pi), \perp/(c, r_0, r_1, x, y, z));$$

14. *Result(π)*: Having the entire history of the privacy game up to the moment of this query, \mathcal{B} can answer faithfully whether the tag was authenticated by the reader or not, or there is another case outside of these. In Q , it will include

$$(Result(\pi), 1/0/\perp),$$

depending on the case;

15. *Corrupt(T)*: The tag T has no permanent variables, its only global temporary variables being c and x . Through corruption its PUF is destroyed. As a result, if the tag is not in \mathcal{T} , \mathcal{B} has nothing to return to \mathcal{A} . Otherwise, it returns the global temporary variables c and x , which \mathcal{A} has learned from previous communications anyway. \mathcal{B} moves then T from \mathcal{T} into the list $c\mathcal{T}$ of corrupted tags. The pair

$$(Corrupt(T), \emptyset/(c, x))$$

is included in Q (depending on the case).

16. *CreateInsider(ID)*: \mathcal{B} creates a new tag reference T , associates it with ID , returns T to \mathcal{A} , and includes (T, ID) in $i\mathcal{T}$. Moreover,

$$(CreateInsider(ID), T)$$

is included in Q .

It is as clear as possible that the probability with which \mathcal{B} guesses to which component, left or right, C applied the function F is precisely the probability with which \mathcal{A} guesses with which tag, left or right, played the privacy game for the Σ scheme. Therefore, the assumption that the protocol is not strongly private will contradict the pseudo-randomness of F . So, the protocol must be strongly private. \square

6 CONCLUSIONS

Analyzing many authentication protocols based on RFID highlights that their privacy properties are established by ad hoc techniques or informally. Trying to do this analysis uniformly in a general model like the HPVP model, it was found that many of these protocols do not ensure privacy. We thus identified two major problems that lead to the lack of privacy of these protocols; we rigorously demonstrated this and abundantly exemplified both techniques. We then focused on a protocol based on simulatable PUFs and showed that a simplified version of it achieves strong privacy in the HPVP model.

Apart from those in Section 4, other scenarios that can lead to a low degree of privacy or even its absence are those in (Hristea and Țiplea, 2020; Țiplea, 2022). A detailed analysis of them is undoubtedly necessary.

REFERENCES

- Adeli, M., Bagheri, N., Sadeghi, S., and Kumari, S. (2023). χ perbp: a cloud-based lightweight mutual authentication protocol. *Peer Peer Netw. Appl.*, 16(4):1785–1802.
- Aghili, S. F., Mala, H., Kaliyar, P., and Conti, M. (2019). SecLAP: Secure and lightweight RFID authentication protocol for medical IoT. *Future Generation Computer Systems*, 101:621–634.
- Armknrecht, F., Sadeghi, A.-R., Scafuro, A., Visconti, I., and Wachsmann, C. (2010). Impossibility results for RFID privacy notions. In Gavrilova, M. L., Tan, C. J. K., and Moreno, E. D., editors, *Transactions on Computational Science XI*, pages 39–63. Springer-Verlag, Berlin, Heidelberg.
- Azad, S. and Ray, B. (2019). A lightweight protocol for RFID authentication. In *2019 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, pages 1–6.

- Tiplea, F. L. (2022). Narrow privacy and desynchronization in Vaudenay's RFID model. *International Journal of Information Security*, 22:563–575.
- Tiplea, F. L., Andriesei, C., and Hristea, C. (2021). Security and privacy of PUF-based RFID systems. In *Cryptography - Recent Advances and Future Developments*. IntechOpen. ISBN 978-1-83962-566-4.
- Tiplea, F. L., Hristea, C., and Bulai, R. (2022). Privacy and reader-first authentication in Vaudenay's RFID model with temporary state disclosure. *Comput. Sci. J. Moldova*, 30(3):335–359.
- Delvaux, J. (2017). Security analysis of PUF-based key generation and entity authentication.
- Fan, K., Jiang, W., Li, H., and Yang, Y. (2018). Lightweight RFID protocol for medical privacy protection in IoT. *IEEE Transactions on Industrial Informatics*, 14(4):1656–1665.
- Fan, K., Luo, Q., Zhang, K., and Yang, Y. (2020). Cloud-based lightweight secure rfid mutual authentication protocol in iot. *Information Sciences*, 527:329–340.
- Fan, K., Zhu, S., Zhang, K., Li, H., and Yang, Y. (2019). A lightweight authentication scheme for cloud-based RFID healthcare systems. *IEEE Network*, 33(2):44–49.
- Gao, Y., van Dijk, M., Xu, L., Yang, W., Nepal, S., and Ranasinghe, D. C. (2022). TREVERSE: TRial-and-Error lightweight secure ReVERSE authentication with simulatable PUFs. *IEEE Transactions on Dependable and Secure Computing*, 19(1):419–437.
- Gope, P. and Sikdar, B. (2021). A comparative study of design paradigms for PUF-based security protocols for iot devices: Current progress, challenges, and future expectation. *Computer*, 54(11):36–46.
- Hermans, J., Pashalidis, Andreas and Vercauteren, F., and Preneel, B. (2011). A new RFID privacy model. In Atluri, V. and Diaz, C., editors, *Computer Security – ESORICS 2011*, pages 568–587, Berlin, Heidelberg. Springer Verlag.
- Hermans, J., Peeters, R., and Preneel, B. (2014). Proper RFID privacy: Model and protocols. *IEEE Transactions on Mobile Computing*, 13(12):2888–2902.
- Hristea, C. and Tiplea, F. L. (2020). Privacy of stateful RFID systems with constant tag identifiers. *IEEE Transactions on Information Forensics and Security*, 15:1920–1934.
- Jin, C., Xu, C., Zhang, X., and Zhao, J. (2015). A secure RFID mutual authentication protocol for healthcare environments using elliptic curve cryptography. *J. Medical Syst.*, 39(3):24.
- Katz, J. and Lindell, Y. (2020). *Introduction to Modern Cryptography*. Chapman & Hall/CRC, 3rd edition.
- Kumar, A., Singh, K., Shariq, M., Lal, C., Conti, M., Amin, R., and Chaudhry, S. A. (2023). An efficient and reliable ultralightweight RFID authentication scheme for healthcare systems. *Computer Communications*, 205:147–157.
- Paise, R.-I. and Vaudenay, S. (2008). Mutual authentication in RFID: Security and privacy. In *Proceedings of the 2008 ACM Symposium on Information, Computer and Communications Security*, ASIACCS '08, pages 292–299, New York, NY, USA. ACM.
- Rührmair, U. and van Dijk, M. (2013). PUFs in security protocols: Attack models and security evaluations. In *2013 IEEE Symposium on Security and Privacy*, pages 286–300.
- Sadeghi, A.-R., Visconti, I., and Wachsmann, C. (2010a). Enhancing rfid security and privacy by physically unclonable functions. In Sadeghi, A.-R. and Naccache, D., editors, *Towards Hardware-Intrinsic Security: Foundations and Practice*, pages 281–305, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Sadeghi, A.-R., Visconti, I., and Wachsmann, C. (2010b). PUF-enhanced RFID security and privacy. In *Workshop on secure component and system identification (SECSI)*, volume 110.
- Safkhani, M., Rostampour, S., Bendavid, Y., and Bagheri, N. (2020). IoT in medical and pharmaceutical: Designing lightweight RFID security protocols for ensuring supply chain integrity. *Computer Networks*, 181:107558.
- Sipser, M. (2012). *Introduction to the Theory of Computation*. Cengage Learning.
- Vaudenay, S. (2007). On privacy models for RFID. In *Proceedings of the Advances in Cryptology 13th International Conference on Theory and Application of Cryptology and Information Security*, ASIACRYPT'07, pages 68–87, Berlin, Heidelberg. Springer-Verlag.
- Wang, Y., Liu, R., Gao, T., Shu, F., Lei, X., Gui, G., and Wang, J. (2023). A novel RFID authentication protocol based on a block-order-modulus variable matrix encryption algorithm.
- Xiao, H., Alshehri, A. A., and Christianson, B. (2016). A cloud-based RFID authentication protocol with insecure communication channels. In *2016 IEEE Trustcom/BigDataSE/ISPA*, pages 332–339.
- Xiao, L., Xu, H., Zhu, F., Wang, R., and Li, P. (2020). SKINNY-based RFID lightweight authentication protocol. *Sensors*, 20(5).
- Xu, H., Ding, J., Li, P., Zhu, F., and Wang, R. (2018). A lightweight RFID mutual authentication protocol based on physical unclonable function. *Sensors*, 18(3).
- Zhu, F., Li, P., Xu, H., and Wang, R. (2020). A novel lightweight authentication scheme for RFID-based healthcare systems. *Sensors*, 20(17).
- Tiplea, F. L. (2022). Lessons to be learned for a good design of private RFID schemes. *IEEE Transactions on Dependable and Secure Computing*, 19(4):2384–2395.
- Tiplea, F. L. and Hristea, C. (2021). PUF protected variables: A solution to RFID security and privacy under corruption with temporary state disclosure. *IEEE Transactions on Information Forensics and Security*, 16:999–1013.