Journal of Engineering Science *Vol. XXXIII, no. 2 (2026), pp. 71 - 84*
Fascicle *Electronics and Computer Science* *ISSN 2587-3474*
Topic *Computers and Information Technology* *eISSN 2587-3482*

https://doi.org/10.52326/jes.utm.2026.33(2).06
UDC 621.396.663:004.056:003.26

LIGHTWEIGHT RFID MUTUAL AUTHENTICATION PROVIDING UNTRACEABILITY

Rodica Bulai *, ORCID: 0009-0003-1898-2575

Technical University of Moldova, 168 Stefan cel Mare blvd., MD2004, Chisinau, Republic of Moldova
* Corresponding author: Rodica Bulai, *rodica.bulai@ati.utm.md*

Received: 04.16.2026
Accepted: 05.27.2026

Abstract. The widespread deployment of low-cost RFID tags raises significant security and privacy concerns, since tags with limited computational resources cannot support strong cryptographic primitives, while backward and forward untraceability have become essential requirements for modern RFID protocols. We hypothesize that a lightweight authentication scheme based on pseudo-random number generators (PRNGs) and short-term session keys can provide a broad spectrum of security properties without relying on costly primitives. The purpose of this study is to design and analyze such a mutual authentication protocol for RFID systems. Two variants of the protocol are constructed - one optimized for limited tag computational power and one minimizing data transmission - using one-time pad encryption keyed by PRNG output, timestamps for replay-attack protection, and a refreshable forward/backward key sub-chain. The resulting protocol withstands replay, denial-of-service, tag and server impersonation attacks, and guarantees information privacy, location privacy, and both backward and forward untraceability. We conclude that even weak primitives such as linear or non-linear feedback shift registers can be safely used as the underlying PRNG, which makes the protocol particularly well suited to low-cost RFID deployments.

Keywords: *attack, traceability, computer security, cryptography, privacy, protocol, RFID system.*

Rezumat. Utilizarea pe scară largă a etichetelor RFID de cost redus generează probleme semnificative de securitate și confidențialitate, deoarece etichetele cu resurse de calcul limitate nu pot susține semnale criptografice puternice, iar imposibilitatea de urmărire a devenit o cerință esențială pentru protocoalele RFID moderne. A fost formulată ipoteza, că o schemă de autentificare de tip ușor, bazată pe generatoare de numere pseudo-aleatorii (PRNG) și chei de sesiune pe termen scurt poate oferi un spectru larg de proprietăți de securitate fără a recurge la primitive costisitoare. Scopul lucrării a constat în proiectarea și analiza unui astfel de protocol de autentificare mutuală pentru sisteme RFID. Sunt construite două variante ale protocolului - una optimizată pentru etichete cu putere de calcul redusă, iar cealaltă pentru minimizarea volumului de date transmise - folosind cifrare cu cheie unică (one-time pad) generată de PRNG, marcaje de timp pentru protecția împotriva atacurilor de tip replay și un lanț reînnoibil de sub-chei înainte/înapoi. Protocolul rezultat rezistă la atacuri replay, de refuz al serviciului, de impersonare a etichetei și a serverului, asigură confidențialitatea informației și a localizării, precum și imposibilitatea de urmărire înainte și

înapoi. S-a constatat că registrele cu deplasare cu reacție liniară sau neliniară pot fi utilizate în siguranță ca funcție PRNG subiacentă, ceea ce face protocolul deosebit de convenabil pentru implementări RFID de cost redus.

Cuvinte cheie: *atac, trasabilitate, securitate informatică, criptografie, confidențialitate, protocol, sistem RFID.*

1. Introduction

Radio Frequency Identification (or RFID for short) has developed at a fast pace, as different security properties emerged from the need to defend against a wave of attacks on deployed protocols. Replying to a reader query with the identifier of the tag began to be viewed as an invasion of privacy due to the possibility of obtaining sensitive information through back-door channels, and was soon replaced with a masking of some sort of the reply. Location traceability became a problem allowing unauthorized readers to track persons based on replies to bogus queries. As in most security protocols, replay attacks and denial of service attacks were considered as threats as soon as the complexity of the protocol increased and the adversary model was granted more power.

Many identification and authentication protocols for RFID systems have been proposed so far. One distinction between them is usually made by the computational power of the tag. Tags with good computational capabilities can implement protocols based on strong cryptographic primitives, such as public-key schemes built on elliptic curve cryptography [1], [2], [3] or symmetric schemes based on AES [4]. More recent constructions rely on physically unclonable functions (PUFs) [5], [6], [7], [8] and have been analyzed in connection with formal privacy models [9]; a broader classification of RFID authentication approaches is given in [10]. Such designs, however, can be too costly to be adopted in most retailer operations which are envisioned as major applications of the RFID technology. A large amount of research has therefore been dedicated to designing RFID authentication protocols based on hash functions and hash function chains [11], [12], [13], as well as on pseudo-random functions and random number generators [14], [15], [16].

Lightweight and ultra-lightweight authentication protocols for RFID systems only require to perform primitive operations such as random number generation, arithmetic bit-wise operations, cyclic redundancy code checksum, or even light hash or pseudo-random functions. Representative ultra-lightweight constructions include the EMAP, M2AP and Gossamer protocols [17], [18], [19], as well as minimalist and human-assisted approaches [20], [21], [22]. More recent proposals target specific application domains such as healthcare, the Internet of Vehicles and mobile commerce [23], [24], [25], [26], or build on contemporary block-cipher primitives [27]. Critical analyses of the security and traceability properties of such schemes are provided in [28], [29], [30], [31]. There is a widespread view that the lightweight and ultra-lightweight authentication protocols will be the best candidate technology for securing the future low-cost RFID systems.

In the rest of the paper we shall focus on lightweight RFID systems. Most low-cost tags have no protection of memory, allowing a strong adversary to read the entire state of the tag at some point, effectively cloning it (given the opportunity). As such, backward traceability, or the ability of an adversary to deduce past states of the tag based on the current state, became an issue, and protocols were designed to resist such attacks by updating the tag's internal state in a manner reproducible by the server and synchronizing upon a successful authentication. One of the most recent security concerns was formally introduced

and studied in [14], namely forward traceability. Intuitively, one would desire an adversary to be unable to deduce future states of the tag based on the present one. Although it is unclear whether this concern was raised by the need for tags to securely change hands (perfect ownership transfer), forward untraceability and perfect ownership transfer were introduced together in [14], along with a protocol that provides them under an assumption.

Security models have been proposed for powerful RFID systems that can classify the level of privacy obtained by them [32], [33]. Recent extensions of these models address adversaries with temporary state disclosure capabilities [34]. Unfortunately, such formal models are missing for (ultra-) lightweight RFID systems. This is why many papers propose lightweight RFID schemes for which security and privacy issues are studied ad hoc, as in [35], [36], [37]. A structured and rather detailed discussion on security concerns in lightweight RFID protocols can be found in [15].

Structure of the paper This paper is structured into six sections. The present section is an introductory one, while the following overviews some related work and draws up our contributions. Two variants of the protocol are described in the third section and their privacy and security properties are inferred in the fourth. Prospective applications for the protocol are analyzed in the fifth section. Conclusions are drawn in the final section..

2. Contributions and Related Work

In this paper we propose a lightweight mutual authentication protocol for RFID, based on a generic pseudo-random number generator (PRNG) and short-term (or rather session) keys. The protocol we propose enjoys a large spectrum of security properties, ranging from protection against low-level attacks (replay attacks and DoS) to attacks mountable by strong adversaries (backward and forward traceability), under a set of assumptions. Another characteristic of the protocol is that it tries to alleviate the computational requirements imposed on the tag as much as possible, shifting most responsibilities to the server.

Moreover, the generic PRNG used in our protocol can be instantiated in various ways, depending on the computational capabilities of the tag and the probabilistic security a deployer desires. For instance, one could use lightweight hash functions, block ciphers, linear feedback shift registers (LFSR) or non-linear feedback shift registers (NLFSR), all resulting in the same level of security considering ideal primitives.

Instead of relying on complex cryptographic primitives, most of the messages exchanged in the protocol we propose are the result of the application of a one-time pad encryption, with keying material provided by a generic pseudo-random number generator. A similar approach was suggested by [38], where a pseudo-random number generator was used together with a shared key for sequence synchronization. The protocol did not take into consideration forward traceability concerns and the data on the tag was static, allowing an adversary that would recover the shared key from the tag to monitor all future communication.

Location privacy and backward untraceability are a direct consequence of the use of a forward key chain for encrypting communication from the tag to the server. Similar approaches were suggested in [13], [14]. Unlike similar approaches, which relied on the server being able to recognize the key chain used to encrypt data and identify the tag, we chose to use the links in our forward key chain (which are basically chained outputs of a pseudo-random number generator) for encryption only, using a temporary or session key to identify the tag, leading to faster queries and eliminating the need for the server to keep a copy of

the forward key chain, reducing storage requirements. The session key is refreshed on each successful authentication to prevent forward traceability.

In order to prevent forward traceability, our protocol used a similar technique to that presented in [14], which adopted a backward key chain, but instead of precomputing the whole chain before deployment, small fragments of a chain (or rather sub-chains) are used and refreshed on each successful authentication, along with the tag identifier, the seed for the pseudo-random number generator and the session key. This refreshing of the tag's state represents a probabilistic evolution, a complementary operation to updating said state in a deterministic way. Similar techniques were used in [14], [15], [39]. Communication from the server to the tag is encrypted using links of the backward key sub-chain, recognizable by the tag and the old session key.

In order to prevent desynchronization, our protocol retains, for each tag, both the values known to be on the tag on the last successful authentication and the new values generated to prevent forward traceability. This doubling up of the authentication data is a common feature for RFID protocols, best represented in the work of [12], [14], [15], [40].

Replay attacks are thwarted by the introduction of timestamps. While other protocols [14], [15] suggest the use of nonces, in an attempt to unburden the tag from any time-consuming computations, we chose to use a timestamp generated by the reader and that will be later checked by the server.

In fact, many features of the protocol were motivated by the desire to minimize the tag's required capabilities. Instead of providing a fresh seed that would be later used by the tag to extract various new elements, all elements are computed on the server and provided to the tag. Similarly, a single pseudo-random function is assumed to be computable. Note that, given a proper relaxation of the notion of pseudo-random number generator or pseudo-random function, one could deploy instances of the protocol described in this paper by using non-linear feedback shift registers (or NLFSRs for short) or even linear feedback shift registers (LFSRs). Although many authors advise against LFSRs due to the ease of their analysis, considering that the seed is constantly being refreshed and the outputs are used exclusively as keying material, it is the authors' belief that deploying an instance based on LFSRs might be considered appropriately secure. Nevertheless, the model does allow for stronger pseudo-random functions to be deployed, and the choice of a particular function should be carefully made to comply with required security parameters. In a sense, this paper introduces a family of identification protocols, instances of which might be suitable for a wide range of applications.

The use of NLFSRs was considered for RFID as far as 2003 [41], and pseudo-random number generators were used as a primary security mechanism for many protocols, including [16], [38], [42], although this frequently lead to weaknesses such as those exposed in [43]. For a more detailed discussion on RFID protocols in general, the reader is referred to [44].

3. Using PRNGs and Short-term Keys for RFID

The protocol we propose comes in two variants, the former requiring less computational power from the tag, while the latter minimizes data transmission.

Notation and nomenclature Throughout the remainder of this section, some terms and notation conventions will be used. A *chain* refers to a sequence generated by a PRNG, while a *sub-chain* is used to denote a continuous part of a chain of significantly reduced dimension.

Link is used to denote a single element in either a chain or sub-chain. Conventional notations that we use are enumerated below:

- $x << k$ denotes the circular left shift of the bit string x by k positions;
- $x \oplus y$ is used to denote the bitwise exclusive or of the bit strings x and y;
- $f^0(x)$ is the identity function;
- $f^{k+1}(x) = f(f^k(x))$, for all $k \geqslant 0$.

3.1. A Protocol for Limited Tag Computational Power

The description of our protocol follows three steps: basic assumptions, initialization, and the authentication protocol.

Assumptions. In the construction of our protocol, we assume the following:

- The server and the trusted readers communicate through a secure channel;
- The reader and the tag communicate via an unsafe channel (wireless, radio frequency channel) that can suffer from eavesdropping or interference from a malicious third party;
- An intruder could attempt to impersonate the tag in order to misdirect an honest reader;
- An intruder could attempt to impersonate a reader to obtain information from a tag;
- An intruder could attempt to impersonate the server while impersonating a reader by responding with a message constructed to resemble a valid reply from the server;
- An intercept-then-invalidate attack is highly unlikely to be successful;
- A strong adversary could obtain the state of a tag at a given moment.

The enumeration above describes the setting taken into consideration when designing the protocol below. To summarize, the third party in the communication (the server) is considered trusted by default and in turn will only reply to trusted readers, reducing the problem of the authentication process to the reader-tag communication.

Initialization. Upon deployment, one should assume that the server has access to a cryptographic hash function (denoted from here on out by h) and a pseudo-random number generator (denoted by f) which would require a seed of truly random bits. This pseudo-random number generator could also be a linear or a non-linear feedback shift register considering an appropriate relaxation of the notion.

The important aspect regarding f is that given a random seed it can produce keying material for a one-time pad and that its output is seemingly random. The function f is also expected to be easily computable by each tag.

The server maintains a database of tag information, some of which would be present on the tag itself and could be recognized upon receiving a relayed query from the reader. For each tag, the server stores the following values:

- Two identification tokens denoted ID_{old} and by ID_{new};
- Two tag identifiers denoted $TID_{old} = h(ID_{old})$ and $TID_{new} = h(ID_{new})$;
- Two session keys denoted SK_{old} and SK_{new};
- Two backward key sub-chains, comprised of the following links: $f^i(BKL_{old})$ and $f^i(BKL_{new})$ for all $0 \leqslant i \leqslant 4$;
- Some data relevant to the tag, denoted *Data*, which does not play any role in the authentication process.

A tag is expected to store the following data:

- A tag identifier denoted *TID*;
- A session key denoted *SK*;
- A forward key link denoted *FKL* (which constitutes the seed for the function *f* to generate keying material);
- A terminal backward key link denoted *TBKL*.

The correlation between tag and server data is enforced by the following process. When registering a tag, the vendor generates *ID*, computes *TID = h(ID)*, chooses at random a session key *SK* and stores the last two on the tag and all of them in the database in ID_{new}, TID_{new} and SK_{new} respectively. It then randomly chooses a backward key link *BKL* and stores it in BKL_{new} iterates the function *f* four times, storing the resulting values in their respective place in the new backward key sub-chain. The fifth application of the f function generates a value to be stored on the tag in the *TBKL* variable. The *FKL* variable on the tag is also initialized with a random value.

Authentication. The following steps describe the communication process in this protocol. For simplicity, denote the tag by T, the reader by R and the server by S.

1. R queries T by transmitting a timestamp τ.

2. T responds by transmitting to R the message $M_1=SK \oplus f(FKL)$, $M_2=f(f(FKL) \oplus \tau) \oplus TID$ At the end of this step, T updates *FKL* to $f^2(FKL)$.

3. R appends τ to the message received from the tag and forwards it to S via a secure channel.

4. Upon receiving the above message, S performs the following actions:

(a) Check freshness of the message by comparing τ to local time.

(b) For some row, extract $\alpha_{new} = SK_{new} \oplus M_1$ and $\beta_{new} = f(\alpha_{new} \oplus \tau) \oplus M_2$

If $\beta_{new} = TID_{new}$, then move the data from the new columns to the old ones and continue to sub-step (c). Otherwise, extract $\alpha_{old} = SK_{old} \oplus M_1$ and $\beta_{old} = f(\alpha_{old} \oplus \tau) \oplus M_2$

If $\beta_{old} = TID_{old}$, then continue to sub-step (c). Otherwise, if there are rows remaining, perform the current sub-step on the following row. If no more rows exist, halt and send R an error message.

(c) T being identified (by the values marked by either new or old), S does the following

- Chooses a random *ID'* and stores it in ID_{new}
- Computes *TID'* = *h*(*ID'*) and stores it in TID_{new}
- Chooses a random session key *SK'* and stores it in SK_{new}
- Chooses a random forward key link *FKL'*
- Chooses a random backward key link *BKL'*, computes $f^i(BKL')$ for all $0 \leqslant i \leqslant 4$, and stores the resulting values in the corresponding columns of the backward key sub-chain marked by new
- Computes the messages $M_3 = SK_{old} \oplus BKL_{old}$, $M_4 = f(BKL_{old}) \oplus TID'$, $M_5 = f^2(BKL_{old}) \oplus SK'$, $M_6 = f^3(BKL_{old}) \oplus FKL'$, $M_7 = f^4(BKL_{old}) \oplus f^5(BKL')$
- Sends M_3, M_4, M_5, M_6, M_7, *Data* to R

5. Upon receiving the message from S, R extracts *Data* and forwards M_3, M_4, M_5, M_6, and M_7 to T.

6. Upon receiving the message from R, T extracts $\gamma = SK \oplus M_3$ and computes $f^5(\gamma)$. If $f^5(\gamma) \neq TBKL$, the tag halts. Otherwise, T refreshes its state as follows:

- Stores $f(\gamma) \oplus M_4$ in the variable *TID*.
- Stores $f^2(\gamma) \oplus M_5$ in the variable *SK*.

- Stores $f^3(\gamma) \oplus M_6$ in the variable *FKL*.
- Stores $f^4(\gamma) \oplus M_7$ in the variable *TBKL*.

This concludes the description of the proposed protocol.

3.2. Minimizing Data Transmission

Slightly raising the computational capabilities of the tag yields a modified protocol that minimizes the number of messages exchanged between parties. For brevity, we choose to enumerate only the changes that have to be made to the protocol in order to minimize data transmission.

1. In the initialization process, the two backward key sub-chains stored in the server's database are comprised of the links $f^i(BKL_{old})$ and $f^i(BKL_{new})$ for $0 \leqslant i \leqslant 2$, while the tag will hold $TBKL = f^3(BKL_{new})$

2. In the authentication process:

(a) Sub-step (c) of the fourth step is replaced by:

- Choose a random backward key link *BKL'*, compute $f^i(BKL')$ for all $0 \leqslant i \leqslant 2$, and store the resulting values in the corresponding columns of the backward key sub-chain marked by new
- Choose a random *ID* and store it in ID_{new}
- Compute $TID' = h(ID')$ and store it in TID_{new}
- Compute $SK' = f(f^3(BKL') << 1)$ and store it in SK_{new}
- Compute M_3, M_4 and M_5 by $M_3 = SK_{old} \oplus BKL_{old}$, $M_4 = f(BKL_{old}) \oplus TID'$, $M_5 = f^2(BKL_{old}) \oplus f^3(BKL')$
- Send *M3*, *M4*, *M5*, *Data* to R

(b) The fifth step becomes: Upon receiving the message from S, R extracts *Data* and forwards M_3, M_4 and M_5 to T.

(c) The final step becomes: Upon receiving the message from R, T extracts $\gamma = SK \oplus M_3$ and computes $f^3(\gamma)$. If $f^3(\gamma) \neq TBKL$, the tag halts. Otherwise, T refreshes its state as follows:

- Stores $f(\gamma) \oplus M_4$ in the variable *TID*.
- Stores $f^2(\gamma) \oplus M_5$ in the variable *TBKL*.
- Stores $f(TBKL << 1)$ in the variable *SK*.
- Stores $f(SK << 1)$ in the variable *TBKL*.

Throughout this description, the circular shifting to the left by one position can be replaced with any simple combination of shifting, splitting and recombination that the tag and the server agree upon and can reproduce separately. The first version of the protocol requires less computations to be done by the tag at the cost of two extra messages on the confirmation step, while at the same time providing strong randomness to the refreshed data. In turn, the second version reduces the number of messages on the confirmation step at the cost of the tag having to perform an extra operation (or set of operations) to refresh its internal state. Depending on the implementation, the fact that three separate variables on the tag are derived from the same source of supposedly truly random bits in a deterministic way might expose the second version to some attacks.

4. Security and Privacy

We discuss in this section the basic security properties our protocol enjoys.

Information privacy. Typically, a tag would respond to a query by providing an identifier that would further be used by an authorized reader to extract more information from the database. Although this strategy might be suitable for most situations, when dealing with

sensitive data it poses some concerns. For instance, an unauthorized reader might make use of some back-channel or some other source of information to obtain data on the tag bearer. The protocol described in the previous section avoids this by encrypting the identifier with a session key shared with the server before replying to the reader.

Location privacy. Another possible threat posed to a tag bearer is unauthorized location tracking. If, somehow, an unauthorized reader can correlate responses to queries to some tag identity, then it can successfully track said tag without having to access server data. This type of attack is thwarted in our protocol by the use of a pseudorandom number generator as a source of keying material for one-time pad encryption of messages exchanged between parties, at the same time allowing the server to recover the seed of the sequence due to the fact that it possesses the session key.

Tag impersonation. An intruder might try to impersonate a tag in a communication with a reader. However, a weak intruder has a negligible chance of issuing a valid response to a query without knowing the current session key. This does not change the fact that a strong intruder might be able to clone a tag by completely mimicking its internal state.

Replay attack. The tag has no way of checking the freshness of the timestamp τ issued by the reader performing the query (maintaining current time is somewhat of a stretch for RFID tag technology and practically infeasible for tags without a battery). However, the timestamp does reach the server and will be subsequently checked (initially upon retrieval) and verified for tampering (by the combination of the first and second part of the relayed message), protecting the server from replay attacks.

DoS attack. A denial of service attack in this scenario would aim to permanently desynchronize the server and the tag, by changing the internal state of the tag to be unrecognizable by the server or by tampering with the terminal backward key link in order to make the server unrecognizable for the tag.

Due to the unlikeliness of the success of an intercept-then-invalidate attack on the confirmation step of the authentication process and considering the assumption that the channel between the reader and the server is secure, an adversary could mount such an attack by gaining knowledge of both the backward key sub-chain leading to the terminal backward key link on the tag and the current session key. Supposing the session key and the terminal backward key link could be obtained, the backward key sub-chain itself must be computed by inverting the pseudo-random function used for generating it. The computational hardness of inverting said function renders this attack impractical.

Even so, an intruder capable of cloning a tag can desynchronize the authentic one successfully by either sequential impersonations leading to database updates or a single successful authentication, followed by a server impersonation due to the knowledge of the backward key subchain inferred from the reply.

Backward traceability. Due to the fact that on every successful authentication the tag's internal state is completely refreshed, one would expect to be able to trace a tag backward at most between two successful authentications. Even supposing the entire internal state is recovered, tracing the tag's action backwards is a challenging attack. Knowledge of the current seed for the pseudo-random function could represent an advantage, especially in the case of linear and non-linear feedback shift registers, which are periodical. An attacker could cycle through an entire period to discover past values of the root and trace back the failed transactions since the last authentication. This data should not be considered relevant, as

failed transactions are most likely results of the actions of an intruder, rendering the gained knowledge useless.

Forward traceability. Supposing an intruder manages to recover the internal state of a tag and instead of impersonating it to desynchronize it from the database decides to track it. This strategy is feasible, as long as the intruder does not miss a single successful authentication, which would result in a complete refresh of the tag's internal data. If he observers the entire process, he can deduce the new state of the tag by following the protocol. Missing the final step of a successful authentication leads to a blind update of the tag's data from the intruder's perspective.

Server impersonation. Suppose an intruder gained access to the internal state of a tag, it could attempt to impersonate the server in order to desynchronize it permanently. One of the attacks is described in a previous paragraph, the other is sketched below.

For linear and non-linear feedback shift registers, and intruder could take advantage of the periodic nature of the function, computing forward values until reaching the starting point again. Doing this for the TBKL variable and already knowing the tag's state, the intruder would basically possess enough information to impersonate the server and possibly desynchronize the tag. A costly operation, making this attack highly unlikely to succeed. Also note that an attacker would have a small window of opportunity for deploying this attack, namely before a successful authentication, in which case the internal state of the tag would be completely refreshed.

5. Prospects

This section introduces some prospective applications of our general protocol, considering different instantiations of the pseudo-random number generator used as a cryptographic primitive. As previously pointed out, this generic notion can take on multiple interpretations as a result of various interpretations. While strong security might require the use of block ciphers or hash functions to simulate this randomness, weaker security demands make room for a wider array of cryptographic primitives, including linear and non-linear feedback shift registers.

These are of particular interest to RFID technology due to their efficient hardware implementation and seemingly simple computation. In light of these arguments, the authors believe a discussion on the advantages and disadvantages of each approach is in order. From now on, two categories will be considered - the strong primitives and the weak primitives, depending on the computational effort an adversary has to put in to invert the underlying function of the pseudo-random number generator.

An underlying function for a PRNG will be considered *a strong primitive* if it requires the same amount of effort for each step backward. That is to say, an adversary attempting to invert the underlying function could not gain a consistent computational advantage by trying to guess a value further back in the sequence. Intuitively, this would imply the presence of some probabilistic choice one would have to make at each backward step.

Alternatively, an underlying function for a PRNG will be considered *a weak primitive* if a single backward step would require virtually as much effort as any fixed number of steps backward. That is to say, an adversary attempting to trace the PRNG backwards would gain a significant advantage by trying to guess values further back in the sequence. Intuitively, this would imply that the underlying function is injective.

5.1. Strong primitives

Strong primitives are best represented by hash functions. The fact that they map a given set to a significantly smaller one makes backward traceability infeasible even in the absence of a refresh from the server. Their impact on an adversary possessing both the session key and the terminal backward key link trying to impersonate the server is limited. However, an adversary in possession of said secret material can easily recover the whole state of the tag effectively cloning it, exposing two other available DoS attacks described in the previous section, making this advantage arguably unimportant.

Implications on backward traceability. Even considering an adversary possessing an oracle for the inverse of a hash function, said adversary would still be faced with the problem of choosing the appropriate former value used as the seed out of the set of values in the inverse. Given a function that hashes inputs of $n + t$ bits to outputs of n bits, an adversary possessing an oracle for the inverse would have to make a choice between $O(2^t)$ values given a somewhat uniformly distributed hash function. One problem with using hash functions is that a single application would not yield such results, as the outputs would have to be extended to $n + t$ bits to be further used either as keying material or as the seed for the following values in the forward key chain. Instead, $1 + \left\lceil \frac{t}{n} \right\rceil$ consecutive applications could be combined in a deterministic way, reproducible by the server, to output a sequence of appropriate length. This, in turn, would complicate things further for an adversary trying to recover past states of the tag, consolidating backward untraceability.

Impact on performance. Having to perform multiple iterations of the same function for a single forward step in the key chain has a massive performance impact on the tag. The overall authentication scheme becomes slower and harder to implement on tags with limited resources. A reasonable alternative for more powerful tags, the authors would however advise against such deployments, as a discussion on backward traceability in the previous section argued that any data recoverable would be limited to the failed communications spanning to the last successful authentication, which by themselves could be considered irrelevant and possibly compromised data, as failed authentications are a strong indicator of outside interference.

5.2. Weak primitives

Weak primitives are injective by contrast to strong primitives. Considering ideal primitives, trying to guess the previous key link based on the current one is as hard as trying to guess any prefix of the current link in the chain. Weak primitives that could successfully be deployed as underlying functions for the PRNG in our protocol include block ciphers, LFSRs and NLFSRs. Arguably, these primitives could be used in the construction of a hash function, yielding strong primitives. The authors, however, advise against this, as no significant advantage would be gained.

Implications on backward traceability. As previously sketched during the discussion on backward traceability from the previous section, given a weak primitive, an adversary could successfully recreate the past states of the tag spanning to the last successful authentication. Note that block ciphers are also periodical due to the fact that they are bijective and that their domain is the same as their co-domain, so the same argument works for block ciphers as well.

Impact on performance. The fact that a single iteration of the underlying function yields the following number in the PRNGs sequence is a great improvement to the previous model

with respect to performance, as all computations are greatly reduced when generating keying material and seeds for future iterations of the PRNG. It is the authors' opinion that our protocol encourages the deployment of weak primitives.

6. Conclusions

Backward and forward untraceability are fundamental security properties for RFID protocols, given that most low-cost tags have no way of securing data stored on them, allowing a strong adversary to read their entire internal state.

The first protocol to provide backward and forward untraceability, along with a mechanism for perfect ownership transfer and other desirable security properties, is the one in [14]. It is based on the OSK protocol [13], which at the time performed rather well, allowing a maximum number of failed authentications that were backward untraceable based on a precomputed forward key chain. The idea in [14] is to refresh the tag data through some form of shared randomness involved in every successful transaction. To avoid desynchronization, data is doubled up in the backend database by retaining both newly generated values and the last known values to be held by the tag. Also, to thwart server impersonation attempts, a backward key chain is used to authenticate the server to the tag in the confirmation step, rendering a mutual authentication protocol. Three pseudo-random functions are considered, and implementations are suggested based on block ciphers rather than hash functions to allow cheaper deployments on RFID tags.

Most influential to our paper was the work in [15], which, besides having a properly structured identification of possible threats (concerning both privacy and security) addressed some performance issues regarding [14] and other prominent protocols at the time [12], [16], [38], [40]. The protocol in [15] assures both information and location privacy, while at the same time covering some of the attacks a weak adversary might perform (tag impersonation, replay attacks, denial of service) and stronger attacks, which assume an adversary capable of cloning tags (backward and forward traceability and server impersonation). The security properties are obtained under some assumptions much like the ones we made regarding the environment in which our protocol should be deployed.

We provided two variants of a protocol for RFID mutual authentication using generic PRNGs. Both variants enjoy desirable security properties such as resilience against replay and DoS attacks, server and tag impersonation and backward and forward traceability. The major privacy concerns in RFID (information and location privacy) are also covered. As to the two variants of the protocol, the former is designed to work with tags having a limited computational power, while the latter would take advantage of the increased computational power of a tag to minimize data transmission.

The generic PRNG can be instantiated in various ways, all classifiable as either weak or strong primitives. Most representative for strong primitives are hash functions, while block ciphers, LFSRs and NLFSRs are considered equally weak in our analysis. Furthermore, based on our findings, we can infer that our protocol encourages the use of weak primitives, as stronger ones offer no significant advantage.

Acknowledgments: The author is grateful to Prof. Ferucio Laurentiu Ţiplea and Prof. Ion Bolun for fruitful discussions on the security and privacy of RFID authentication protocols, which have shaped the ideas presented in this work.

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

References

1. Batina L, Guajardo J, Kerins T, et al (2006) An elliptic curve processor suitable for RFID tags. https://www.researchgate.net/publication/220337092_An_elliptic_curve_processor_suitable_for_RFID-tags. Accessed 16 April 2026
2. Batina L, Guajardo J, Kerins T, et al (2007) Public-key cryptography for RFID-tags. In: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops. pp 217–222
3. Lee YK, Sakiyama K, Batina L, Verbauwhede I (2008) Elliptic-curve-based security processor for RFID. IEEE Trans Comput 57:1514–1527. https://doi.org/10.1109/TC.2008.148
4. Feldhofer M, Dominikus S, Wolkerstorfer J (2004) Strong authentication for RFID systems using the AES algorithm. In: Joye M, Quisquater J-J (eds) Cryptographic Hardware and Embedded Systems – CHES. Lecture Notes in Computer Science, vol 3156. Springer, pp 357–370
5. Ţiplea FL, Andriesei C, Hristea C (2020) Security and privacy of PUF-based RFID systems. In: Cryptography - Recent Advances and Future Developments. IntechOpen, London, pp 1–23. https://doi.org/10.5772/intechopen.94018
6. Ţiplea FL, Hristea C (2021) Practically efficient RFID scheme with constant-time identification. In: Proceedings of the 18th International Conference on Security and Cryptography (SECRYPT), Vol. 1. SciTePress, pp 495–506. https://doi.org/10.5220/0010544804950506
7. Ţiplea FL, Hristea C (2021) PUF protected variables: a solution to RFID security and privacy under corruption with temporary state disclosure. IEEE Trans Inf Forensics Secur 16:999–1013. https://doi.org/10.1109/TIFS.2020.3027147
8. Hristea C, Ţiplea FL (2020) Privacy of stateful RFID systems with constant tag identifiers. IEEE Trans Inf Forensics Secur 15:1920–1934. https://doi.org/10.1109/TIFS.2019.2953398
9. Ţiplea FL (2022) Narrow privacy and desynchronization in Vaudenay's RFID model. Int J Inf Secur 21:563–575. https://doi.org/10.1007/s10207-021-00569-0
10. Pantelić G, Bojanić S, Tomašević V (2009) Authentication protocols in RFID systems. In: Zhang Y, Kitsos P (eds) Security in RFID and Sensor Networks. Auerbach Publications, pp 99–120
11. Avoine G, Oechslin P (2005) A scalable and provably secure hash-based RFID protocol. In: Third IEEE International Conference on Pervasive Computing and Communications Workshops. pp 110–114
12. Henrici D, Muller P (2004) Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers. In: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops. pp 149–153
13. Ohkubo M, Suzuki K, Kinoshita S (2003) Cryptographic approach to "privacy-friendly" tags. In: RFID Privacy Workshop.https://www.researchgate.net/publication/2926255_Cryptographic_Approach_to_Privacy-Friendly_Tags. Accessed 16 April 2026
14. Lim CH, Kwon T (2006) Strong and robust RFID authentication enabling perfect ownership transfer. In: Ning P, Qing S, Li N (eds) Information and Communications Security. Lecture Notes in Computer Science, vol 4307. Springer, pp 1–20
15. Song B, Mitchell CJ (2008) RFID authentication protocol for low-cost tags. In: Proceedings of the First ACM Conference on Wireless Network Security. pp 140–147
16. Weis SA, Sarma SE, Rivest RL, Engels DW (2004) Security and privacy aspects of low-cost radio frequency identification systems. In: Hutter D, Müller G, Stephan W, Ullmann M (eds) Security in Pervasive Computing. Lecture Notes in Computer Science, vol 2802. Springer, pp 201–212
17. Peris-Lopez P, Hernandez-Castro JC, Estevez-Tapiador JM, Ribagorda A (2006) EMAP: an efficient mutual-authentication protocol for low-cost RFID tags. In: Meersman R, Tari Z, Herrero P (eds) On the Move to Meaningful Internet Systems. Lecture Notes in Computer Science, vol 4277. Springer, pp 352–361
18. Peris-Lopez P, Hernandez-Castro JC, Estevez-Tapiador JM, Ribagorda A (2006) M2AP: a minimalist mutual-authentication protocol for low-cost RFID tags. In: Ma J, Jin H, Yang LT, Tsai JJ-P (eds) Ubiquitous Intelligence and Computing. Lecture Notes in Computer Science, vol 4159. Springer, pp 912–923
19. Peris-Lopez P, Hernandez-Castro JC, Estevez-Tapiador JM, Ribagorda A (2009) Advances in ultralightweight cryptography for low-cost RFID tags: Gossamer protocol. In: Chung K-I, Sohn K, Yung M (eds) Information Security Applications. Springer, pp 56–68
20. Juels A (2005) Minimalist cryptography for low-cost RFID tags (extended abstract). In: Blundo C, Cimato S (eds) Security in Communication Networks. Lecture Notes in Computer Science, vol 3352. Springer, pp 149–164

21. Juels A, Weis SA (2005) Authenticating pervasive devices with human protocols. In: Shoup V (ed) Advances in Cryptology – CRYPTO. Lecture Notes in Computer Science, vol 3621. Springer, pp 293–308
22. Vajda I, Buttyán L (2003) Lightweight authentication protocols for low-cost RFID tags. In: Second Workshop on Security in Ubiquitous Computing – Ubicomp 2003. https://www.researchgate.net/publication/2930099_Lightweight_Authentication_Protocols_for_Low-Cost_RFID_Tags. Accessed 16 April 2026
23. Chander B, Gopalakrishnan K (2022) A secured and lightweight RFID-tag based authentication protocol with privacy-preserving in telecare medicine information system. Comput Commun 191:425–437. https://doi.org/10.1016/j.comcom.2022.05.002
24. Khorasgani AA, Sajadieh M, Yazdani MR (2022) Novel lightweight RFID authentication protocols for inexpensive tags. J Inf Secur Appl 67:103191. https://doi.org/10.1016/j.jisa.2022.103191
25. Kumar S, Banka H, Kaushik B, Sharma S (2021) A review and analysis of secure and lightweight ECC-based RFID authentication protocol for internet of vehicles. Trans Emerg Telecommun Technol 32:e4354. https://doi.org/10.1002/ett.4354
26. Mala H, Aghili SF (2019) Security analysis of an ultra-lightweight RFID authentication protocol for m-commerce. Int J Commun Syst 32:e3837. https://doi.org/10.1002/dac.3837
27. Xiao L, Xu H, Zhu F, et al (2020) SKINNY-based RFID lightweight authentication protocol. Sensors 20:1366. https://doi.org/10.3390/s20051366
28. Alavi SM, Baghery K, Abdolmaleki B, Aref MR (2015) Traceability analysis of recent RFID authentication protocols. Wirel Pers Commun 83:1663–1682. https://doi.org/10.1007/s11277-015-2469-0
29. Chien HY, Huang CW (2007) Security of ultra-lightweight RFID authentication protocols and its improvements. SIGOPS Oper Syst Rev 41:83–86
30. Chien HY, Huang CW (2010) A lightweight authentication protocol for low-cost RFID. J Signal Process Syst 59:95–102
31. Piramuthu S (2007) Protocols for RFID tag/reader authentication. Decis Support Syst 43:897–914. https://doi.org/10.1016/j.dss.2007.01.003
32. Hermans J, Peeters R, Preneel B (2014) Proper RFID privacy: model and protocols. IEEE Trans Mob Comput 13:2888–2902. https://doi.org/10.1109/TMC.2014.2314127
33. 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. Springer, Berlin, Heidelberg, pp 68–87
34. Ţiplea FL, Hristea C, 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
35. Burmester M, De Medeiros B (2008) The security of EPC Gen2 compliant RFID protocols. In: Proceedings of the 6th International Conference on Applied Cryptography and Network Security. Springer, Berlin, Heidelberg, pp 490–506
36. Ţiplea FL (2014) A lightweight authentication protocol for RFID. In: Kotulski Z, Ksiezopolski B, Mazur K (eds) Cryptography and Security Systems – Third International Conference. Communications in Computer and Information Science, vol 448. Springer, pp 110–121
37. Năstase GD, Ţiplea FL (2015) On a lightweight authentication protocol for RFID. In: Bica I, Naccache D, Simion E (eds) Innovative Security Solutions for Information Technology and Communications – SECITC. Lecture Notes in Computer Science, vol 9522. Springer, pp 212–225
38. Molnar D, Wagner D (2004) Privacy and security in library RFID: issues, practices, and architectures. In: Proceedings of the 11th ACM Conference on Computer and Communications Security. pp 210–219
39. Dimitriou T (2005) A lightweight RFID protocol to protect against traceability and cloning attacks. In: First International Conference on Security and Privacy for Emerging Areas in Communications Networks. pp 59–66
40. Chien HY, Chen CH (2007) Mutual authentication protocol for RFID conforming to EPC Class 1 Generation 2 standards. Comput Stand Interfaces 29:254–259. https://doi.org/10.1016/j.csi.2006.04.003
41. Weis SA (2003) Security and privacy in radio-frequency identification devices. MS Thesis, Massachusetts Institute of Technology, Cambridge, MA
42. Duc DN, Park J, Lee H, Kim K (2006) Enhancing security of EPCglobal Gen-2 RFID tag against traceability and cloning. In: Proc SCIS, Hiroshima, Japan. p 97
43. Yeh KH, Lo NW (2010) Improvement of two lightweight RFID authentication protocols. Inf Assur Secur Lett 1:6–11. https://www.mirlabs.org/iasl/volume_1/IASL_Vol_1_Paper_2.pdf. Accessed 16 April 2026

44. Zhang Y, Kitsos P (2009) Security in RFID and Sensor Networks, 1st edn. Auerbach Publications, Boston. https://www.researchgate.net/publication/326990799_Security_in_RFID_and_sensor_networks. Accessed 16 April 2026

Citation: Bulai, R. (2026). Lightweight RFID mutual authentication providing untraceability. *Journal of Engineering Science*. 2026, 33 (2), pp. 71-84. https://doi.org/10.52326/jes.utm.2026.33(2).06.

Submission of manuscripts: jes@meridian.utm.md