<table>
<thead>
<tr>
<th>Title</th>
<th>A privacy-preserving efficient RFID authentication protocol from SLPN assumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author(s)</td>
<td>Islam Mamun, Mohammad Saiful, Miyaji, Atsuko</td>
</tr>
<tr>
<td>Citation</td>
<td>International Journal of Computational Science and Engineering, 10(3): 234-243</td>
</tr>
<tr>
<td>Issue Date</td>
<td>2015</td>
</tr>
<tr>
<td>Type</td>
<td>Journal Article</td>
</tr>
<tr>
<td>Text version</td>
<td>author</td>
</tr>
<tr>
<td>URL</td>
<td><a href="http://hdl.handle.net/10119/12946">http://hdl.handle.net/10119/12946</a></td>
</tr>
<tr>
<td>Description</td>
<td></td>
</tr>
</tbody>
</table>
A privacy-preserving efficient RFID authentication protocol from SLPN assumption

Mohammad Saiful Islam Mamun* and Atsuko Miyaji

Japan Advanced Institute of Science and Technology (JAIST),
1-1, Asahidai, Nomi, Ishikawa 923-1292, Japan
E-mail: mamun@jaist.ac.jp
E-mail: miyaji@jaist.ac.jp
*Corresponding author

Abstract: This paper presents an authentication protocol of RFID system where both the tag and the reader are authenticated mutually. Optimal performance requirement, considering storage and computation constraints of low-cost tags, keeping security and privacy policies intact are some major challenges in recent research in this area. We propose a secure and private mutual authentication protocol of HB-family to meet the demand of low-cost tags. It is composed of subspace learning parity from noise (SLPN) problem and pseudo-inverse matrix properties where both of them significantly reduce the cost in terms of computation and hardware requirements. In addition, we compare our protocol with other existing HB-like and ordinary RFID authentication protocols according to their construction primitives and security and privacy achievements.

Keywords: RFID; mutual authentication; privacy; subspace learning parity from noise; SLPN problem; pseudo-inverse matrix.


Biographical notes: Mohammad Saiful Islam Mamun is a Doctoral candidate at Information Security Lab, School of Information Science, JAIST, Japan. He received his BSc (Hons.) in Computer Science and Engineering from Dhaka University, Bangladesh and MS in Information and Communication System Security from Royal Institute of Technology (KTH), Sweden in 2005 and 2008, respectively. His primary research interests include applied cryptography and information system security.

Atsuko Miyaji received her BSc, MSc and Dr. Sci. in Mathematics from Osaka University, Osaka, Japan in 1988, 1990, and 1997, respectively. She joined Panasonic Co., Ltd. from 1990 to 1998 and engaged in research and development for secure communication. She has been a Professor at the Japan Advanced Institute of Science and Technology (JAIST) since 2007. Her research interests include the application of number theory into cryptography and information security. She is a member of the International Association for Cryptologic Research, the Information Processing Society of Japan, and the Mathematical Society of Japan.

1 Introduction

Tag authentication is an indispensable approach to prevent an RFID tag from impersonation. In particular, tag authentication is more significant since tags are much vulnerable to counterfeit than readers. However, mutual authentication protocols add an additional protection for the RFID system in the protocol construction to safeguard the query is, in fact, coming from a legitimate reader.

Unlikability or untraceability, sometimes referred to interchangeable with same meaning, conveys the property that an adversary cannot distinguish whether two events occurring in an RFID system are related to the same tag or not. In addition, anonymity is another indispensable security property that assure the inability to identify a tag within an RFID system. This definition can be framed in terms of unlikability by saying that a tag is anonymous in any transactions between the reader provided that adversary cannot link the tag to a transaction. In order to provide aforementioned security properties, ample research has been done in this area targeting enhanced privacy, security and performance issues. Since asymmetric key ciphers are too expensive for a compact hardware such as low-cost RFID tag, majority of the authentication protocols use symmetric key as secret. For example, RSA require more than 30,000 gates, which is too expensive for low-cost tag where maximum 2,000 gates out of 10,000 gates are available for the purpose of security (Juels and Weis, 2005).

The LPN problem is a light-weight provably-secure cryptographic scheme which was first introduced in 2001 by Hopper and Blum (2001). LPN-based authentication is not only theoretically secure in terms of provable security, but also provides better efficiency than classical symmetric
ciphers that are not related to hard problems. There has been a large body of research on HB protocol that outputs protocols such as HB, HB’ , HB’’ , HB-TP, HB-TP’, HB’’, etc. (Katz et al., 2010; Gilbert et al., 2005, 2008a; Bringer et al., 2006; Munilla and Peinado, 2007; Ouafi et al., 2008; Leng et al., 2008). Unfortunately, all of them later shown to be insecure or susceptible to particular attacks (Gilbert et al., 2008b; Ouafi et al., 2008). In Pietrzak et al. (2011), authors propose an authentication protocol based on the subspace learning parity from noise (SLPN) problem with tight security reduction which is as efficient as the previous HB-family, but has twice the key length; in addition, their proof works in quantum setting, which leads the protocol to be secure against quantum adversaries.

To the best of our knowledge, the latest addition to the HB-family for RFID authentication is F-HB, where authors use two LPN problems as their basic computation (Cao and ONeill, 2011). We carefully observe that the Toeplitz matrix multiplication (EX-OR operation) for the multiple bit LPN problem and MAC generation in the main protocol of Cao and ONeill (2011) are not consistent with matrix size, although the authors did not clarify the specific matrix size in operation; and the threshold value for LPN problem is not specified concretely. Moreover, in the last protocol transcripts, where a tag’s secret key is updated, if-checking, is not consistent and is not based on the LPN problem; but an EX-OR vector computation. Unlike Cao and ONeill (2011), our protocol follows the SLPN-based problem for tag authentication, where the secret key is not a vector but a binary matrix. In addition, we introduce pseudo-inverse matrix for updating the secret key of the tag and apply to the SLPN problem for both the tag and the reader authentication. As a consequence, our proposed protocol is more robust against quantum adversaries while been efficient like the previous HB-protocol family.

The rest of this paper is organised as follows. Section 2 introduces notations and assumptions used in this paper and other useful definitions related to basic primitives and security notions. The proposed protocol is described in Section 3. In Section 4, all achieved security and privacy attributes are discussed in detail with their corresponding proof; while Section 5 covers the analysis and comparison results. Finally, Section 6 concludes this paper.

## 2 Preliminaries

In this section, we first briefly introduce the notations used in the paper in Table 1. Then we discuss some inevitable assumptions followed by useful definitions for primitives and security notions.

### 2.1 Assumption

We assume the RFID system described in this paper consist of a single legitimate reader and a set of tags (EPC global Class-1 Generation-2). The reader is connected to the backend server that stores all the relevant data including the tag database. Initially, the reader generates and set \( T_d \) and \( S_t \) public parameters depending on security parameter \( \lambda \). Each tag has its unique identification \( T_d \) and session key \( S_t \). \( T_d \) is used as the shared secret key between the tag and the reader. The authentication protocol is an interactive protocol executed between tags/prover and a reader/verifier where both are probabilistic polynomial time (PPT) algorithms. All communications between the server and the reader are assumed to be secure and over an authentic channel. For simplicity, we consider the reader and server as identical. Throughout the paper, we use the term reader and server interchangeably. A tag is not a tamper-resistant device; so its session key \( S_t \) is refreshed after each session is completed successfully. For updating the key, the tag authenticates the reader first. An adversary cannot compromise the reader/server and cannot corrupt the tag until it compromises both \( T_d \) and \( S_t \) at the same time. However, if both of the secret keys are exposed at a time, the adversary can trace the tag for a certain period \( i \) until the next authentication cycle starts. We assume tag binary identification \( T_d \) is unique within an RFID system. To avoid an exhaustive database search at the reader, hash-index (I) is used. Database at the server associates the tag index with other tag-related data, e.g., \( T_d \), \( S_t \), \( P_t \), etc.

<table>
<thead>
<tr>
<th>Table 1</th>
<th>Notations used in this paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \lambda )</td>
<td>Security parameter</td>
</tr>
<tr>
<td>( \mathbb{Z}_p )</td>
<td>Set of integers modulo an integer ( p \geq 1 )</td>
</tr>
<tr>
<td>( l \in \mathbb{N} )</td>
<td>Length of the secret key</td>
</tr>
<tr>
<td>( n \in \mathbb{N} )</td>
<td>Number of parallel repetitions ( n \leq b/2 )</td>
</tr>
<tr>
<td>( T_d )</td>
<td>2/ bit EPC or unique ID of a tag</td>
</tr>
<tr>
<td>( I_i )</td>
<td>( n ) index of the tag during time period ( i )</td>
</tr>
<tr>
<td>( P_i )</td>
<td>( l \times n ) bit matrices as session key for the reader during time period ( i )</td>
</tr>
<tr>
<td>( S_i )</td>
<td>( l \times n ) bit matrices as session key between the reader and the tag during time period ( i )</td>
</tr>
<tr>
<td>( s )</td>
<td>( 2/ ) bit vector random binary number generated by the reader</td>
</tr>
<tr>
<td>( s' )</td>
<td>( 2/ ) bit vector random binary number generated by the tag</td>
</tr>
<tr>
<td>( w(s) )</td>
<td>Hamming weight of the vector ( s )</td>
</tr>
<tr>
<td>( \tau )</td>
<td>Parameter of the Bernoulli error distribution ( Ber ), where ( r \in [0, 1/2] )</td>
</tr>
<tr>
<td>( \tau' )</td>
<td>Authentication verifier acceptance threshold (tag/reader) where ( \tau' = 1/4 + \tau/2 )</td>
</tr>
<tr>
<td>( e )</td>
<td>( n ) bit vector from Bernoulli distribution ( Ber ), with parameter ( r ), ( Pr[e = 1] = r )</td>
</tr>
<tr>
<td>( [Q] )</td>
<td>( l \times n ) bit non-singular binary matrices randomly generated by the reader</td>
</tr>
<tr>
<td>( [S]^T )</td>
<td>Transpose of matrices ([S]) i.e., ( T : \mathbb{Z}_2^{\tau l} \rightarrow \mathbb{Z}_2^{l n} )</td>
</tr>
<tr>
<td>( [P]^+ )</td>
<td>Pseudo-inverse of a matrices ([P])</td>
</tr>
<tr>
<td>( (x \downarrow y) )</td>
<td>The vector derived from ( x ) by deleting all the bits ( x[i] ) where ( y[i] = 0 )</td>
</tr>
<tr>
<td>( \oplus, | )</td>
<td>Bitwise XOR operation and concatenation of two vectors respectively</td>
</tr>
</tbody>
</table>
2.2 Definitions for primitives

Definition 1: A protocol is called \((t, Q, \epsilon)\)-hard if there exist a PPT adversary \(\mathcal{A}\), usually called \((Q, t)\)-adversary that makes \(Q\)-queries in running time \(t\) to the honest prover, has an advantage at most \(\epsilon\),

\[ \Pr[\mathcal{A} \text{ succeeds}] - \frac{1}{2} \leq \epsilon \]

Definition 2: Let \([R] \in \mathbb{Z}_2^{\tau \times n}\), \(s \in \mathbb{Z}_2^\tau\), \(\tau\) be the noise parameters, and \(e \in \mathbb{Z}_2^n\) be selected from Ber, s.t., \(w(e) \leq tl\). Given \(r = ([R] \cdot s) \oplus e \in \mathbb{Z}_2^\tau\), we denote \(\text{LPN}(\tau, l)(s)\) for the distribution \(\mathbb{Z}_2^\tau \leftarrow \mathbb{Z}_2^\tau \times \mathbb{Z}_2^n\).

The decisional LPN problem is \((t, Q, \epsilon)\)-hard to distinguish uniform random binary vectors \(U_l\) from \(\text{LPN}(\tau, l)(s)\) with random secret \(s \in \mathbb{Z}_2^\tau\),

\[ \text{Adv}^\text{LPN}_{\mathcal{A}}(\tau, l) = \Pr[s \in \mathbb{Z}_2^\tau : \text{LPN}(\tau, l)(s) = 1] - \Pr[U_l : \text{LPN}(\tau, l)(s) = 1] \leq \epsilon \]

Definition 3: The SLPN problem is defined as a biased half-space distribution where the adversary can ask not only with secret \(s\) but also with \(r \cdot s \oplus e'\), where \(e', r\) can be adaptively chosen with sufficient rank(r). Let \(s \in \mathbb{Z}_2^\tau\) and \(l, n \in \mathbb{Z}_2\) where \(n \leq l\). The decisional SLPN problem is \((t, Q, \epsilon)\)-hard such that,

\[ \text{Adv}^\text{SLPN}_{\mathcal{A}}(\tau, l, n) = \Pr[\text{LPN}^*_{\tau, l, n}(s, \gamma) = 1] - \Pr[U_l : \text{LPN}^*_{\tau, l, n}(s, \gamma) = 1] \leq \epsilon \]

Definition 4: The subset LPN problem (SLPN*) is defined as a weaker version to SLPN problem where the adversary cannot ask for all inner products with \(r' \cdot s \oplus e';\) for any rank(r') \(\geq n\) but only with subset of \(s\). Let \((l, n, v) \in \mathbb{Z}_2\) where \(n \leq l\) and \(w(v) \geq n\) where \(v\) can be adaptively chosen. Hence, \(\text{LPN}^*_{\tau, l, n}(s, v)\) samples are of the form \(([R] \cdot v) \oplus e\) and \(\text{LPN}^*_l(v)\) takes \(v\) as input and output a sample of \(U_l\). The SLPN* problem is \((t, Q, \epsilon)\)-hard such that,

\[ \text{Adv}^\text{SLPN}^*_{\mathcal{A}}(\tau, l, n) = \Pr[\text{LPN}^*_{\tau, l, n}(s, \gamma) = 1] - \Pr[U_l : \text{LPN}^*_{\tau, l, n}(s, \gamma) = 1] \leq \epsilon \]

Definition 5: In linear algebra, a pseudo-inverse \(A^+\) of a matrix \(A\) is a generalisation of the inverse matrix. The most widely known and popular pseudo-inverse is the Moore-Penrose pseudo-inverse, which was independently described by Moore (1920). An algorithm for generating pseudo-random matrix on non-singular matrix \(\mathbb{Z}_2\) is given in Thuc et al. (2010). However, the matrix \(A\) is the unique matrix that satisfies the following properties:

- \(AA^+A = A\)
- \(A^+AA^+ = A^+\)
- \((A^+A)^T = A^+A\)
- \((A^+)^+ = A\)

2.3 Definitions for security notions

Definition 6: A protocol is secure against passive attacks, if there exists no PPT adversary \(\mathcal{A}\) that can forge the verifying entity with non-negligible probability by observing any number of interactions between the tag and reader.

Definition 7: A \((t, Q, \epsilon)\)-hard protocol is called secure against active attacks where the adversary \(\mathcal{A}\) runs in two stages: First, it observes and interrupts all the interactions between the target tag \(T\) and legitimate reader with concurrent executions according to the defined security. Then, it is allowed only one time to convince the reader. Note that, this time \(\mathcal{A}\) is not allowed to continue his attacks in time instance \(t\), but can utilise several discrete or successive time period.

Definition 8: In the man-in-the-middle (MIM) attack, adversary \(\mathcal{A}\) is allowed to maintain connections with both the tag and the reader, making the tag believe that they are talking directly to the reader over a secure connection, when in fact, the entire communication is controlled by \(\mathcal{A}\). Then, \(\mathcal{A}\) interacts with the reader to authenticate. The goal of the attacker \(\mathcal{A}\) is to authenticate successfully in \(Q\) rounds. \(\mathcal{A}\) is successful if and only if it gets accept response from all \(Q\) rounds.

Definition 9: The forward security property means that even if the adversary obtains the current secret key, it cannot derive the keys used for past time periods.

Definition 10: The backward security is opposite to the forward security. If the adversary can explore the secret of the tag at time \(i\), it cannot be traced in future using the same secret. In other words, exposure of a tag’s secret should not reveal any secret information regarding the future of the tag. But if an adversary is allowed to obtain full access to the tag’s secret, and thus can trace the target tag at least during the current session of authentication immediately following the attack, it does not make any sense to perfect security in practice. Therefore, it is impossible to provide backward security for an RFID-like device practically.

Definition 11: Tracking a tag refers the attacker could guess the tag identity or link multiple authentication sessions of the same tag. In our protocol, the adversary cannot recover \(S_i\) or any other information identifying that particular tag.

Definition 12: In de-synchronisation attack, the adversary aims to disrupt the key update, leaving the tag and the reader in a desynchronised state and renders future authentication impossible.
Definition 13: Denial of service (DoS) is an attempt to make a tag unavailable to its intended users. DoS resistance capability of the protocol is infinite as tag updates the key after reader authentication is successful.

Definition 14: Tag cloning entails that the data on a valid tag is scanned and copied by a malicious RFID reader, and later the copied data will be embedded onto a fake tag.

Definition 15: In the replay attack, an adversary reuses the communication scripts from the former sessions to perform a successful authentication between each tag and their reader.

Definition 16: An RFID system is said to unconditionally provide privacy notion $X$, if and only if for all adversaries $A$ of type $X$, it holds that $Adv_{A}^{X}(\lambda) = 0$. In case of computational privacy, it is $Adv_{X}^{C}(\lambda) \leq \epsilon$ for all PPT adversaries $A$ (Hermans et al., 2011).

Definition 17: An RFID system is said to be $(Q, t, \epsilon)$ strong private, if there exist no $(Q, t)$ adversary $A$ who can break its strong privacy with advantage $Adv_{A}^{S}(k) \geq \epsilon$.

3 Construction

We adopt the idea of key-insulation to slightly twist our three-round mutual authentication protocol described in Figure 1. The protocol allows significantly less computations to a tag. On the other hand, the most expensive computations of the protocol are handled by the reader. We use only random generation, bitwise XOR and matrix multiplication as tag operation. The protocol uses $(\lambda, \tau, \tau', n, l)$ as public parameters, where $(\tau, \tau')$ are constant while $(l, n)$ depends on the security parameter $\lambda$. For initialisation, the server generates the initial index $I_{0}$, the session key $S_{0}$ and its corresponding $P_{0}$ and other public parameters; and set the necessary data into a tag non-volatile memory. Note that, we use matrix as a secret, not a vector. Therefore, for each tag, there is a tuple $[I, T_{id}, S_{i}, P_{i}, P_{i-1}, P_{]}$ to be stored in the back-end database of the server at any time instance $i$.

<table>
<thead>
<tr>
<th>Reader $(I_{i}, T_{id} \in \mathbb{Z}<em>{2}^{l}, S</em>{i} \in \mathbb{Z}<em>{2}^{l \times n}, P</em>{i} \in \mathbb{Z}_{2}^{l \times T})$</th>
<th>Tag$(I_{i}, T_{id} \in \mathbb{Z}<em>{2}^{l}, S</em>{i} \in \mathbb{Z}_{2}^{l \times n})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbb{Z}_{2}$; where $w(s) = l$</td>
<td>$\mathbb{Z}_{2}$; where $w(s') = l$</td>
</tr>
<tr>
<td>$s \in \mathbb{Z}_{2}$; if $w(s) \neq l$ return;</td>
<td>$s' \in \mathbb{Z}_{2}$; if $w(s') \neq l$ return;</td>
</tr>
<tr>
<td>$e \in \mathbb{Z}<em>{2}$; $r := [S</em>{i}]^{T} \cdot (T_{id} \cdot s) \oplus e$</td>
<td>$r' := [S_{i}]^{T} \cdot (T_{id} \cdot s') \oplus e'$</td>
</tr>
<tr>
<td>$I_{i+1} = r$ if $w(s') \neq l$ return;</td>
<td>$P_{i+1} = [S_{i}]\cdot [S_{i+1}]^T \in \mathbb{Z}_{2}^{l \times l}$</td>
</tr>
<tr>
<td>Generate non-singular $[Q] \in <em>{R} \mathbb{Z}</em>{2}^{n \times l}$</td>
<td>$P_{i+1} = [S_{i}] \cdot [S_{i+1}]^T \in \mathbb{Z}_{2}^{n \times l}$</td>
</tr>
<tr>
<td>$[S_{i+1}] = [Q] \cdot [S_{i}] \in \mathbb{Z}_{2}^{n \times l}$</td>
<td>$e' \in \mathbb{Z}<em>{2}$; $r' := [S</em>{i}]^{T} \cdot (T_{id} \cdot s') \oplus e'$</td>
</tr>
<tr>
<td>where $rank(S_{i+1}) = n$</td>
<td>$P_{i+1} = [P_{i}] \cdot [Q] \in \mathbb{Z}_{2}^{l \times l}$</td>
</tr>
<tr>
<td>Compute $P_{i+1} = [S_{i+1}] \cdot [S_{i+1}]^T \in \mathbb{Z}_{2}^{l \times l}$</td>
<td>$e' \in \mathbb{Z}<em>{2}$; $r' := [S</em>{i}]^{T} \cdot (T_{id} \cdot s') \oplus e'$</td>
</tr>
<tr>
<td>$P_{i+1} = [S_{i}] \cdot [S_{i+1}]^T \in \mathbb{Z}_{2}^{n \times l}$</td>
<td>$r' := [S_{i}]^{T} \cdot (T_{id} \cdot s') \oplus e'$</td>
</tr>
</tbody>
</table>

Lookup $T_{id}$ by using hash-table index:

Direct match: $I = I_{i}$; if (not found) then

Brute-force search: find an entry $[I, T_{id}, S_{i}, S_{i-1}, S_{i}, P_{i-1}, P_{i}]$ s.t., $\exists (S_{i}$ or $S_{i-1}$), for which the following satisfies:

If $w([S_{i}]^{T} \cdot (T_{id} \cdot s) \oplus r) > n.\tau'$ return;
Else

$I_{i+1} = r$
if $w(s') \neq l$ return;
Generate non-singular $[Q] \in _{R} \mathbb{Z}_{2}^{n \times l}$

$[S_{i+1}] = [Q] \cdot [S_{i}] \in \mathbb{Z}_{2}^{n \times l}$
where $rank(S_{i+1}) = n$
Compute $P_{i+1} = [S_{i+1}] \cdot [S_{i+1}]^T \in \mathbb{Z}_{2}^{l \times l}$
where $[S_{i+1}]^T = ([S_{i}]^{T} \cdot [S_{i+1}]) \cdot [S_{i+1}]^T \in \mathbb{Z}_{2}^{n \times l}$
$P_{i+1} = [P_{i}] \cdot [Q] \in \mathbb{Z}_{2}^{l \times l}$
$e' \in \mathbb{Z}_{2}$; $r' := [S_{i}]^{T} \cdot (T_{id} \cdot s') \oplus e'$
if $w([S_{i}]^{T} \cdot (T_{id} \cdot s') \oplus r') > n.\tau'$ return; else accept
$S_{i+1} = (P_{i}', S_{i+1}) \in \mathbb{Z}_{2}^{n \times n}$
if $rank([S_{i+1}]) \neq n$ return;
For tag authentication, let a tag have $S_i$ and $I_i$ which have been derived from the previous $(i-1)$ successful authentication sessions.

- **Reader**: Generate a random binary challenge string $s$, and sends it to a tag.
- **Tag**: Check the *hamming weight* of the string $s$ and generate an $n$-bit noise vector $e$, a random $2l$-bit challenge string $s'$ for a reader with hamming weight $l$. Next, an $n$-bit LPN problem is computed as $r := [S_i]^T \cdot (T_{2l}[s] \oplus e)$. To eliminate brute-force searching at the server end, maintain an index $I_i$, and send it to the reader. Finally, update index $I_{i+1}$ to $r$ and send $(I_{i+1}, s')$ to the server.

- **Reader**: First search database to find a tuple $[I_i, T_{2l}, S_{i+1}, S_i, P_{i+1}, P_i]$ with index $I_i$ sent by the server. But searching might fail sometimes, e.g., due to synchronisation attack, etc. If it fails, then apply brute-force method targeting to explore $S_i$ or $S_{i+1}$ such that it satisfies LPN problem: $w([S_i]^T \cdot (T_{2l}[s] \oplus r)) \leq n \cdot e'$, or $w((S_i^t)^T \cdot (T_{2l}[s] \oplus r)) \leq n \cdot r'$. If the brute-force method passes, it accepts the tag, update the index to $I_{i+1}$, and enter reader authentication phase.

For reader authentication, it has secret $S_i$, $P_i$, and other public parameters which has been derived from previous $(i-1)$ successful authentication sessions.

- **Reader**: First test whether *hamming weight* of $s'$ is exactly $l$. Then generate a non-singular binary matrix $Q$ to update session key $S_{i+1}$ as $[Q \cdot S_i]$ and compute pseudo inverse-matrix $S_{i+1}'$ and $P_{i+1}$ as $[S_{i+1} \cdot S_{i+1}']$. To send the new session key $S_{i+1}$ to the tag and blinding the matrix $Q$, $P_i'$ is computed by $[P_i \cdot Q]$ which is actually equivalent to a binary matrix $[S_i S_i/S_i/Q]$. Assume the adversary cannot reveal $S_i$ from $P_i'$ in polynomial time. Next, for reader authentication, generate an $n$-bit noise vector $e'$ and compute multiple bit LPN problem as $r' := [S_i]^T \cdot (T_{2l}[s'] \oplus e')$. Finally answer the tag with string $(P_i', r')$.

- **Tag**: Check the hamming weight of $([S_i]^T \cdot (T_{2l}[s'] \oplus r')) \leq n \cdot r'$ where $(n \cdot r')$ is the predefined accepted threshold value for the LPN problem. If this check passes, accept the reader and update session key $S_{i+1}$ by $[P_i' \cdot S_i] = (S_i Q \cdot S_i) = (S_i Q)$ where $S_i S_i = S_i$. However, if the check fails, tag’s session key remains unchanged.

Note that, in the protocol, session key generated by the reader is used by the tag. To be precise, session key $S_{i+1}$ is generated from the former key $S_i$ and random matrix $[Q]$. Sending $S_{i+1}$ as plain text is not secure since $[S_{i+1}]$ will act as the next session key between the tag and the reader. Therefore, random matrices $[S_{i+1}]$ is sent with encryption to the tag. We first use $[Q]$ for randomising $S_i$ and then pseudo-random matrix computation for blinding the matrix $[S]$. However, a tag’s session key is updated each time period $i$ by computing $S_{i+1}$ from simple decryption using pseudo-inverse matrix properties. More precisely, tag’s session key is not updated until a successful reader authentication.

**Hash-table lookup**: An appropriate lookup hash-function can offer efficient database searching. In our protocol, $index$ is updated in both the tag and the reader, as the transaction becomes successful. This demands an efficient hash-table that provide $O(1)$ query, insertion and deletion operations at high loads. We suggest segmented hash table architecture described in Kumar and Crowley (2005), that provides high collision resistance and comparatively low search cost in worst case performance. A traditional hash table maps the key, e.g., index into a single hash bucket, whereas $N$-segmented hash table maps into $N$ potential buckets. Therefore, a table with capacity $m$ has equally sized logical segments containing $m/N$ buckets. Here, the hash function is defined as $\mathcal{H} : I \rightarrow \{0, 1, \ldots, mN-1\}$ where $I$ is the index space of size $n$. Let *linear chaining* be used as searching technique, then average and worse search time will be $\Theta(1 + \varepsilon)$ and $\Theta(\log n / \log \log n)$ respectively, where $\varepsilon = n/m$. To ensure $O(1)$ searches, they utilise $N$-independent bloom filter to achieve low false positive rates.

### 4 Security analysis

#### 4.1 SLPN problem

We use a proof method similar to that described in Pietrzak et al. (2011) as Theorem 1 follows. Even though the protocol in our model and that in Pietrzak et al. (2011) are different, a similar proof can be used as both are based on the SLPN* problem. The hardness of SLPN* can be defined using an indistinguishability game. More formally, the security of the proof is based on the computational indistinguishability of the two oracles SLPN* and uniform distribution $U_{2n}$. From the protocol description, it can be found that noise is a vector rather than a single bit; and the secret is not a vector but a pseudo-random matrix.

**Theorem 1**: For any constant $\gamma > 0$, let $d = l/(2 + \gamma)$. If the SLPN*($s_\gamma$) problem is $(t, nQ, \varepsilon)$-hard, then the authentication protocol from Figure 1, is $(\gamma', Q, \varepsilon')$-secure against active adversaries, where the constants $(\varepsilon_2, \varepsilon') > 0$ depend only on $\gamma$ and $\varepsilon_2$, respectively.

$$t' = t - poly(\Omega, l) \cdot \varepsilon + \varepsilonQ.2^{-\varepsilon_2 \cdot d} + 2^{-\varepsilon_2 \cdot n} = \varepsilon + 2^{-\theta(n)}$$

The protocol has completeness error $2^{-\varepsilon_2 \cdot n}$ where $\varepsilon_2 > 0$.

**Theorem 2**: Let an oracle be $O$ which is either an SLPN*($s_\gamma$) oracle or $U_{2n}()$ defined in Definition 4. Let $B$ be a simulator that uses $(t, Q, \varepsilon)$-adversary $A$ such that:

$$\Pr[B^{SLPN*(\gamma')} = 1] \geq \varepsilon - Q.\alpha'/d \quad \text{and} \quad \Pr[B^{U_{2n}()} = 1] \leq \alpha'^*_\varepsilon$$

where
\[ a_i' \leftarrow \Pr \left( \|w(l) - w(d)\| \leq 2^{-c_i} \right) \]

and

\[ a_i'' \leftarrow \Pr \left( \|w(r) \leq n \cdot r' : r' \in \mathbb{Z}_2^l \| \leq 2^{-c_i} \right) \]

Therefore, \( B \) can distinguish between two oracles SLPN\(^*(s, \cdot)\) and \( U_d(\cdot) \) with advantage \( \epsilon' = Q \cdot a_i'_d - a_i'' \).

Now we can upper bound the gap between two probability that \( B \) outputs:

\[ \left| \Pr[B^{SLPN^*(s, \cdot)} = 1] - \Pr[B^{U_d(\cdot)} = 1] \right| \leq Q a_i'_d \]

This implies the probability of success of the simulator \( B \), and hence the adversary \( A \), in the indistinguishability game.

Interested readers are referred to Pietrzak et al. (2011), for further clarification and proof of the theorem.

### 4.2 MIM attack

The most sophisticated and realistic attack in an RFID system is the MIM attack. Our protocol is MIM-secure against an active attack from the SLPN assumption. Note that, first the reader authenticates the tag, and then vice versa. In case of tag authentication, it runs a two-round MIM-secure authentication protocol where the reader chooses a random variable as challenge, and tag returns the response according to the challenge. The authentication tag \( \gamma = (S, r : S' \oplus f(s)) \oplus e \), where \( f(s) \) is the secret key derivation function which uniquely encodes challenge \( s \) according to \( k \) by selecting \( l \) bits from the key \( k \). The main technical difficulty to build a secure MIM-free authentication from LPN is to make sure the secret key \( k \) does not leak from verification queries. In Pietrzak et al. (2011), they use randomise-mapping function \( f(s) = (k \downarrow s : \mathbb{Z}_2^l \rightarrow \mathbb{Z}_2^l) \) for some random \( s \) and prove that if LPN is hard, then the construction is MIM-secure. We have twisted a little the original idea. In our construction, we retain both \( S \) and \( k \) secret, that enhances security. We use an EX-OR operation for hiding \( s' \) using \( T_{id} \) as key. Note that, the XOR cipher is vulnerable to frequency analysis; therefore, even if the adversary compromises \( T_{id} \), it cannot generate \( S_i \) for any subsequent sessions using only \( T_{id} \). In the third phase of the protocol, we introduce a pseudo-random matrix as blinding factor to transfer the new session key \( S_{i+1} \), which is secure from the pseudo-random matrix property assumption.

### 4.3 Pseudo-random matrix

We followed the security analysis in Thuc et al. (2010), where it is claimed that, having known the messages \( XX'Q \in \mathbb{Z}_2^{n \times l} \), it is impossible to recover the secrets \( X \in \mathbb{Z}_2^{n \times l} \), or \( Q \in \mathbb{Z}_2^{l \times l} \). Given \( XX'Q \in \mathbb{Z}_2^{l \times l} \), suppose that \( \text{rank}(X) = r \), and

\[ X + X = \begin{pmatrix} I_r^r & 0 \\ 0 & 0 \end{pmatrix} \Rightarrow X'Q \begin{pmatrix} Q^{r \times r} & 0 \\ 0 & 0 \end{pmatrix} \]

where \( I_r^r \) is an identity matrix and \( Q^{r \times r} \) is the left upper sub-matrix of \( Q \). Then the probability that an adversary determines the correct \( Q \) is \( 2^{(l-r)n} \). To ensure security, we need to ensure that \( l >> r \), which can be obtained with \( l > n \).

In our authentication protocol, let \( n \leq l/2 \) to ensure a large value of \( l \).

#### 4.4 Forward security

For each operation, the tag uses session key \( S_i \) and the reader also uses its corresponding \( P_i \) for verification of authentication tags. At the end of each valid session, \( (S_i, P_i) \) is updated with the random matrix and the previous key is deleted permanently in the tag. We say that, even if \( S_i \) is exposed by the attacker during the authentication session \( i \), the tag’s privacy is fully guaranteed for \( (i - 1) \) periods.

#### 4.5 Backward security

Typical RFID tags and their reader communicate only for a short period of time because of the power constraint of a tag. Thus, either we restrict the adversary in such a way that it can obtain neither \( T_{id} \) nor \( S_i \) at any time instance \( i \), or there should exist some non-empty gap between the time of a reveal query and the attack, while the tag is not accessible by the adversary. This entails the adversary miss the protocol transcripts needed to update the compromised secret key and hence our protocol claims reduced backward security.

#### 4.6 Tracking a tag

Protocol can resist tracking the tag due to the following reason: it refreshes the random vector \( (s, s', e, e') \), updates the keys \( (P_i, S_i) \) while assumptions like the SLPN problem, the pseudo-random matrix makes the protocol indistinguishable from the adversarial perspective.

#### 4.7 De-synchronisation attack

We introduce indexing of the tag to get rid of the attack. When the reader and the tag maintain synchronisation, searching hash table becomes very fast with direct match technique. However, synchronisation attack may take place in the third protocol transcript from the reader to the tag; while the tag may not receive \( (p', r') \) to update its shared key. In the later case, brute-force search will be used for successful authentication. Although it yields worse performance, but after successful authentication synchronisation would be recovered.

#### 4.8 Tag cloning

We use two different keys \( T_{id} \) and \( S_i \) for the tag. Therefore, even if the tag is cloned by a malicious reader, we assume either of the keys is not compromised. For instance,
an EPC generation 2 allows a password-enabled secure state configuration that prevents anyone from reading or writing onto a tag memory bank. Let $T_d$ be stored in a password-protected memory bank. Moreover, the tag is not allowed to update the key $S_t$ until it authenticates the reader. This verification thwarts the cloning attack as well.

4.9 Replay attack

Assuming that the random challenges sent by the reader and the tag are the same in two different sessions, an adversary can launch a replay attack by snooping the random numbers; but in our protocol, the reader queries the tag each time with a new random challenge $s$, and then the tag queries the reader with random $s', t$. So, it is very unlikely to find a match between a pair of $(s, t, r)$ from two different sessions of the same tag.

4.10 Privacy

We define oracles according to the following:

- $CTag(\text{ID}) \rightarrow T_d$: On input of a tag identifier, this oracle registers the new tag to the reader/server and return a reference $T_d$ to resist duplicate IDs.

- $Launch() \rightarrow \pi, s$: This oracle launches a new protocol by returning a session identifier $\pi$ and first transcript $s$ by the reader to ensure reader-initiated protocol.

- $DTag(T_i, T_j) \rightarrow vtag$: On input of a tag reference $(T_i, T_j)$, this oracle generates a virtual tag reference $vtag$ and stores the triple $(vtag, T_i, T_j)$ in a table $D$, provided that none of the $(T_i, T_j)$ are already referenced in the table. Depending on the value of the random bit $b$ by the challenger, $vtag$ either refers to $T_i$ or $T_j$.

- $Free(vtag, b) \rightarrow T$: On input of $vtag$, it erases the volatile memory of the tag $T_j(b = 0)$ or $T_j(b = 1)$ and removes the entry $(vtag, T_i, T_j)$ from $D$.

- $SendTag(vtag, s) \rightarrow r'$: On input of $vtag$, this oracle sends $s$ to either $T_j(b = 0)$ or $T_j(b = 1)$. It returns the reply $r'$ of the tag or $\bot$.

- $UKey(S_t) \rightarrow S_{t+1}$: A tag key update oracle performed on the tag side which takes $S_t$ as input and outputs an updated key $S_{t+1}$.

- $SReader(\pi, s') \rightarrow s''$: On input of $(\pi, s')$, this oracle sends $s'$ to the reader in session $\pi$ and returns the reply $s''$ of the reader or $\bot$.

- $Result(\pi)$: This oracle returns either 1 or 0 on successful authentication of a tag. But If the session $\pi$ is not finished, or there exists no session $\pi$ it returns $\bot$.

- $Corrupt(T_j)$: On input of $T_j$, this oracle returns the non-volatile internal state of $T_j$. Note that, corruption is done w.r.t. tag, not the $vtag$. Therefore, the adversary is forced to corrupt tags $T_j$ that are currently not drawn.

First, we analyze our protocol using the privacy model in Hermans et al. (2011). Where challenger runs the $Exp_B^\lambda(S)$ experiments with the above oracles.

- $b \in \{0, 1\}$

- SetupReader($1^k$)

- $b' \leftarrow A^{CTag, Launch,DTag,Free,STag,SReader,Result}$

- return ($b' = b$).

We assume that $A$ queries the challenger with $Exp_B^\lambda(S)$ experiments a number of times and hence guess bit $b'$ and wins the privacy game if and only if ($b' = b$). The advantage of the adversary to win is defined as

$$Adv_B^\lambda(k) = \left| Pr\left[Exp_B^\lambda(k)\right] + Pr\left[Exp_B^\lambda(k)\right] - 1\right|$$

The reader sends out a random vector $s$ and the tag computes the protocol transcript from the challenge $s$, combined with shared key $k_t$ and ($e, [R]$). The reader decrypts the tag’s reply and verifies whether it gets right under the shared key $k_t$ in the database. In the second phase, it encrypts the random matrix $[Q]$ with the session key $P_t$ and computes the protocol transcript from the challenge vector $s'$ sent from the tag under the shared secret key $k$. Tag can decrypt the matrix $[Q]$ with session key $S_t$ and verify $e'$ under the shared secret key $k_t$ and MAC value $s''$.

Theorem 3: If the encryption in the protocol described in Figure 1 is indistinguishable then the protocol is strong private for narrow adversaries.

Proof: We analyze our protocol using the privacy model in Hermans et al. (2011). Given an adversary $A$ that wins the privacy game with non-negligible advantage, we consider another adversary $B$ that can break the indistinguishability game with non-negligible advantage described in Section 4.1. The adversary $B$ runs the adversary $A$ to answer queries with the following exceptions:

- $S, T_d$ are two different keys of the indistinguishability game.

- $SendTag(vtag, s):$ By retrieving the tag $T_j$ and $T_j$ references from the table $D$ using virtual tag $vtag$; it generates two references $m_0 = w([S_t])^T(T_i, s') \oplus r > n.t'$ and $m_1 = w([S_t])^T(T_j, s) \oplus r > n.t'$. The references $m_0$, $m_1$ are sent to the indistinguishability oracle of SLPN problem, which returns whether the hamming weight satisfies $w \leq n.t'$ under one of the references.

- $B$ cannot query for $Result()$ oracle.

At the end of the game, $B$ outputs according to $A$’s guess. Hence, $B$ is perfectly simulated for $A$. If $A$ breaks the privacy, then $B$ wins the indistinguishability game; but indistinguishability with only one call to the oracle is equivalent to indistinguishibility with multiple calls to the oracle that proves the narrow privacy of the protocol.
In Ng et al. (2009), the authors have categorised RFID authentication protocols into four types according to their constructions and distinguished eight privacy levels by their natures on accessing $\text{Corrupt}(\cdot)$ oracle in the strategies of the adversary and whether $\text{Result}(\cdot)$ oracle is used or not.

- **Nil**: No privacy protection at all.
- **Weak**: Adversary has access to all oracles except $\text{Corrupt}(T_i)$.
- **Forward**: Adversary has access to $\text{Corrupt}(T_i)$ but other oracles are not allowed as $\text{Corrupt}(T_i)$ oracles are accessed.
- **Destructive**: No restriction on accessing other oracles after $\text{Corrupt}(T_i)$, but $T_i$ is not allowed to use again.
- **Strong**: It is the strongest defined privacy level with no restrictions.

Each of these levels has its narrow counterpart to restrict the access of $\text{Result}(\cdot)$ oracle. Our protocol belongs to Type 2a for construction where the shared key $S_h$ has been updated just after the reader is authenticated. We now redefine our protocol privacy according to the model described in Ng et al. (2009).

Without reader authentication, any adversary can keep querying a tag with any compatible reader until it is desynchronised with a legitimate reader. Therefore, the tag’s secret can only be desynchronised by one update. As the reader has both the keys $S_r$ and $S_{r,i}$, in case of tag failure to update its shared key $S_r$, the reader can still try to authenticate the victim using the previous key $S_{r,i}$ in the next protocol conversation. Thus, it provides weak privacy to the protocol construction. Let an adversary $\mathcal{A}$ try to send authentication transcripts to the tag by blocking a valid reader authentication message, or by intercepting of the tag in an online attack. This causes the tag to be in a DoS attack or in a deadlock condition, as it cannot update the key without reader authentication.

**Theorem 4**: The protocol described in Figure 1 is weak non-narrow privacy preserved.

Due to lack of space, we remove the proof of the above theorem. That will appear in the full version. However, this narrow-forward privacy level attack can be reduced if tag accepts any value to update the key. We can reduce the protocol to narrow-forward privacy level by two ways. Firstly, by reduced backward security, where we restrict the adversary in such a way that there should exist some non-empty gap between the time of a reveal query and the attack, while tag is not accessible by the adversary; which means the adversary misses the protocol transcripts needed to update the compromised secret key (Song and Mitchell, 2008). Secondly, note that $\text{Corrupt}(\cdot)$ oracle operates w.r.t. a tag not with a virtual tag $\text{vtag}$, which means adversary is forced to corrupt tags $\text{Ti}$ that are currently not drawn. Therefore, after single $\text{Corrupt}(\cdot)$ oracle, henceforth adversary is allowed to use $\text{DrawTag}(\cdot, \cdot)$ oracle. Of course, here adversary is not allowed to access $\text{Result}(\cdot)$ oracle.

**Theorem 5**: Considering aforementioned assumptions (Reduced Backward security or disallowing $\text{Result}(\cdot)$ oracle), the protocol described in Figure 1 is semi-forward narrow privacy preserved.

### 5 Comparison and performance analysis

In order to support dynamic scalability, the proposed protocol requires to search and store the lookup hash table for each transaction, based on the index value in online, to retrieve the corresponding data in the hash-table. However, the data can be pre-computed in the hash-table either in offline or dynamically in online.

In case of the tag, protocol operations include two random binary vector generation, one SLPN problem, one EX-OR operation, and three binary linear matrix multiplications. For computation, we only consider the SLPN problem and assume the rest of the operations (e.g., calculation hamming weight) to be trivial in terms of computational complexity. The protocol is roughly as efficient as the HB+ protocol with just twice the key length. Since it is a reduction of the LPN to the SLPN problem, the protocol is secure against quantum adversaries, assuming LPN is secure against such adversaries. There is a natural trade-off between the communication cost and key size. For any constant $c$ ($1 \leq c \leq n$), the communication cost can be reduced by a factor of $c$ by increasing the key size with the same factor.

Major computations of the proposed authentication scheme on the tag include linear binary matrix multiplication and the LPN problem. And, in case of storage, only a secret key and an index for the key. As bitwise XOR, matrix multiplication, the hamming weight $w(\cdot)$ and $(a,b)$ are all binary operation, they can be implemented using bit-by-bit serialisation to save hardware gates. In the e-STREAM project, the PRNG operation needs only 1,294 gates to achieve 80-bit security level using Grain-v1 (Cid and Robshaw, 2009). A PRNG requires a linear feedback shift register (LFSR) structure to compute, so LPN problem can share the same LFSR. $s'$ can be deduced from the state variable of PRNG. The cost of a LPN problem and of storing the index and secret key may not be greater than that of a PRNG, and should be less than that of a CRC as well. However, the LPN problem can be implemented using an LFSR (for transpose matrix), a 1-bit multiplier plus 1-bit accumulator (for binary multiplication), XOR gates (for $\oplus$ operation), 1-bit counter (for hamming weight) and a 1-bit comparator (for $a \cdot b$ operation). Thus, to achieve a $\lambda$-bit security level, the overall hardware cost of the proposed protocol for the above mentioned functions on a tag is no more than 1,600 gates, including the cost of non-volatile memory to store the secret key, the index value and protocol intermediate values; and the protocol is suitable for Class-1 Generation-2 EPC tags, where PRNG and CRC are used as hardware.
**Tag resources and security comparison with HB family and others**

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Storage</th>
<th>Computation (major)</th>
<th>Authentication</th>
<th>Security achieved</th>
<th>Hardware (gates)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pietrzak et al. (2011)</td>
<td>1S</td>
<td>1 LPN</td>
<td>Tag</td>
<td>1, 4*</td>
<td>≈1,600</td>
</tr>
<tr>
<td>HB (Hopper and Blum, 2001)</td>
<td>1S</td>
<td>1 LPN</td>
<td>Tag</td>
<td></td>
<td>≈1,600</td>
</tr>
<tr>
<td>HB+ (Gilbert et al., 2005)</td>
<td>2S</td>
<td>2 LPN</td>
<td>Tag</td>
<td>7</td>
<td>≈1,600</td>
</tr>
<tr>
<td>HB-MP (Munilla and Peinado, 2007)</td>
<td>2S</td>
<td>1 LPN</td>
<td>Tag</td>
<td>5, 6, 7, 9</td>
<td>≈1,600</td>
</tr>
<tr>
<td>HB-MP+ (Leng et al., 2008)</td>
<td>2S</td>
<td>1 LPN, 1 HASH</td>
<td>Tag</td>
<td>1, 5, 6, 7, 9</td>
<td>≈3,500</td>
</tr>
<tr>
<td>F-HB (Cao and O'Neill, 2011)</td>
<td>1I, 1S</td>
<td>1 PRNG, 2 LPN</td>
<td>Mutual</td>
<td>1, 2, 4*, 5, 6, 7, 9</td>
<td>≈3,500</td>
</tr>
<tr>
<td>Ours</td>
<td>1I, 1S</td>
<td>1 LPN, 1 PIM</td>
<td>Mutual</td>
<td>1, 2, 3*, 4, 5, 6, 7, 8, 9</td>
<td>≈1,600</td>
</tr>
<tr>
<td>Avoine and Oechslin (2005)</td>
<td>1S</td>
<td>1 PRF, 1 HASH</td>
<td>Tag</td>
<td>2, 4, 6, 8, 9</td>
<td>≈6,000</td>
</tr>
<tr>
<td>Le et al. (2007)</td>
<td>1I, 1S</td>
<td>1 PRF</td>
<td>Mutual</td>
<td>2, 4*, 6, 8, 9</td>
<td>≈6,000</td>
</tr>
<tr>
<td>Berbain et al. (2009)</td>
<td>1S</td>
<td>1 PRNG, 1 UH</td>
<td>Tag</td>
<td>2, 4, 9</td>
<td>≈3,500</td>
</tr>
<tr>
<td>Billet et al. (2010)</td>
<td>1S</td>
<td>1 SC</td>
<td>Mutual</td>
<td>2, 4*, 8, 9</td>
<td>≈2,000</td>
</tr>
<tr>
<td>Tsudik (2006)</td>
<td>1S, 2TS</td>
<td>1 HASH</td>
<td>Tag</td>
<td>4*</td>
<td>≈1,500</td>
</tr>
<tr>
<td>Chatmon et al. (2006)</td>
<td>1S, 1TS, 1RN</td>
<td>2 HASH</td>
<td>Mutual</td>
<td>4*, 8, 9</td>
<td>≈1,500</td>
</tr>
<tr>
<td>He et al. (2009)</td>
<td>1RN, 1C, 1TS, 1S</td>
<td>3 HASH</td>
<td>Mutual</td>
<td>2, 4*, 6, 8, 9</td>
<td>≈1,500</td>
</tr>
</tbody>
</table>

Notes: Where SC = stream cipher; S = secret key; C = counter; I = index; PRNG = pseudo random number generator; UH = universal hash; PIM = pseudo inverse matrix; LPN = learning parity from noise; TS = time stamp; RN = random number. Security attributes: MIM attack (1), forward security (2), backward security (3), reduced backward security (3*), high privacy (4), limited privacy (4*) tag tracking (5), de-synchronisation (6), tag cloning (7), replay attack (8), DoS (9).

In Table 2, we show a comparative study on some general attributes, e.g., storage consumption, major computations, authentication party, achieved security, approximate hardware cost, etc., between our protocol and several HB-like and non-HB protocols. It appears that, although the tag’s hardware cost of the proposed protocol is optimal, it achieves most common security requirements. Additionally, it achieves $O(1)$ time complexity during the synchronised state that resists brute-force searching in each authentication session. Alternatively, hardware cost of the reader is expensive for the purpose of complex computing, that results in reduced computing in tag and hence hardware cost. Besides that, the hash-indexed searching technique at the reader, where all the data related to certain tags are stored efficiently as index, reduces an exhaustive database search at the reader end. As a consequence, in an RFID system with remote authentication, reader can use this index in batch mode operation to aggregate responses from several tags together, that reduces the communication cost between the reader and the server, where each tag contains unique index within the reader’s field of view at a specific time instance.

### 6 Conclusions

This paper presents a novel hardware-friendly RFID authentication protocol based on the SLPN problem that can meet the hardware constraints of the EPC Class-1 Generation-2 tags. In comparison to other protocols as described in Table 2, it requires less hardware and has achieved major security attributes. The protocol is also compliant to semi forward for narrow adversaries privacy settings. Moreover, scalability of the protocol can be realised best in synchronised and desynchronised modes that ensures infinite DoS resistance. Security and privacy can be protected as long as we allow an adversary not to cope with both tag ID and the secret key simultaneously. In addition, the security and privacy proof follows the standard model that uses indistinguishability as basic privacy notion. Our future research will focus on how to reduce the communication cost between the reader and server, assuming the wireless link between them is insecure, to figure a realistic privacy-preserving RFID environment.

### Acknowledgements

Research has been partially supported by Graduate Research Programme (GRP), JAIST foundation grants and NTT C&C grants no. 24.016.

### References


Notes

1. Result of noisy inner products of vectors.
2. From the properties of pseudo-inverse matrix.
3. To provide scalability.
4. An on-chip predictive filter that supports space-efficient membership queries.
5. We use $T_{id}$ as the secret key $k$.
6. Searching the database and generating a pseudo-random matrix are the most complex part of the protocol.
7. Tag readers are portable and server access is costly.