Hindawi
Security and Communication Networks
Volume 2019, Article ID 6862052, 15 pages
https://doi.org/10.1155/2019/6862052

Research Article

An Enhanced and Resource-Aware RFID Multitag Grouping Protocol

Hala Abu Dhailah, Eyad Taqieddin, and Abdallah Alma'aitah

Jordan University of Science and Technology, P.O. Box 3030, Irbid 22110, Jordan

Correspondence should be addressed to Eyad Taqieddin; eyadtaq@just.edu.jo

Received 30 November 2018; Revised 7 April 2019; Accepted 24 April 2019; Published 23 May 2019

Academic Editor: Prosanta Gope

Several grouping proof protocols were presented to meet the security requirements of Radio Frequency Identification Systems. Nevertheless, these protocols were shown to be vulnerable to various attacks. In this work, we cryptanalyze one of the newest grouping proof protocols. Through this analysis, we show the weaknesses of the protocol and launch a full-disclosure attack to disclose all secrets in the protocol. We show that the probability of success of the protocol is one and that increasing the length of the strings adds little complexity to the attack. We follow this by proposing an enhanced version of the protocol with better overall security. We show its efficiency by providing a security and performance analysis and comparing it with some of the existing protocols in the literature.

1. Introduction

Radio Frequency Identification (RFID) is a well-known item identification system that has many advantages over traditional systems, such as barcodes [1]. First of all, it uniquely identifies each item through the use of a unique identifier. Second, contrary to barcode systems, it does not need direct line of sight in order to identify the items [2]. These advantages led to the wide deployment of this technology in a verity of applications such as manufacturing, transportation, asset management, retailing, logistics, and medical purposes [3, 4].

Passive tag RFID systems consist of three main components: passive (battery-less) tags, readers, and backend servers (referred to as Backend Processing Systems (BPSs) in this work). Tags are usually attached to the object due to their small footprint. RFID readers identify the tags by transmitting a high power RF signal that is backscattered by the tag. The BPS collects the identities of the tags from the reader(s); herein, the link between the reader(s) and BPS is assumed to be trusted [2].

In many applications, an evidence of simultaneous reading of a given group of tags is required. For instance, in the pharmaceutical sector, evidence that the medicine is sold with its prescription is needed. The yoking proof [5] provides an evidence to a third party (e.g., BPS) that the medicine is sold with its prescription [3]. Another example is the automatic library loan-services where a tag is embedded in the books with another tag embedded in the ID card of a user [6]. Further uses appear in hospitals to generate evidence that a specific patient has been administered his/her drugs [7]. The concept of yoking proof was later extended to become "grouping proof" where more than two tags are involved in generating the proof.

In general, grouping proof protocols can be divided into two categories depending on the way that the reader collects the grouping proof: reading order-dependent and reading order-independent grouping proof protocols. For reading order-dependent grouping proof protocols, the reader generates the proof by creating chains between the tags, sending messages from one tag to the next within the tags present in the group [8]. Hence, a tag participating in the proof cannot start its computations until after the preceding tag in the group has already responded to the reader. On the other hand, in reading order-independent grouping proof protocols, the reader broadcasts a message to all tags that participate in the proof. Later, the tag receives the message from the reader and starts executing the needed computations without waiting for

any response from other tags. Although it incurs lower delays, this type is more vulnerable to attacks such as privacy attacks and replay attacks [9].

Grouping proof protocols are executed in two modes: online and offline depending on the existence of the BPS [10]. In the online mode, the BPS can send and receive messages with tags (via the reader). In contrast, the BPS in offline mode cannot unicast any message to the tags when generating the proof but, rather, sends a challenge to the reader. Grouping proof in the online mode is usually close to RFID mutual authentication protocols [6] while offline mode is more challenging as the BPS is not present during the proof generation.

Due to the inherent security vulnerabilities in wireless communications, several attacks have been launched against RFID systems. Some of the targets of these attacks are mutual authentication, ownership transfer, and grouping proof protocols. These attacks are divided into passive and active attacks. In the former, the attackers eavesdrop on the exchanged messages between the reader and tags. These are stored for later offline analysis with the goal of revealing the secret information (e.g., *ID* and keys) [11]. The attackers in this type neither interact with tags nor the reader.

As for the active attacks, various malicious actions are done in addition to those in passive attacks [12]. The attackers can either block or intercept the messages, or modify the exchanged messages.

The most common attacks against RFID systems are the full disclosure [13, 14], desynchronization [15], and tag/reader impersonation [16].

Although numerous protocols for securing RFID communications were proposed by the research community, most of them were found to be vulnerable to some or all of the common attacks mentioned earlier. Their weakness stemmed from poorly implemented message exchanges or from employing weak cryptographic functions.

Recently, Shen et al. proposed a protocol targeted towards supporting grouping proofs [17]. The protocol is efficient in terms of message exchanging overhead and computation complexity at the reader and the tags. It is claimed to be secure against various known attacks.

In this work, however, we show that a full-disclosure attack is feasible against Shen et al. protocol (henceforth referred to as the GAMTP). We show the inherent properties of the GAMTP that render it vulnerable to the full-disclosure attack and the steps needed to successfully execute it and retrieve all the secret values. We further propose an enhancement to the protocol to overcome these weaknesses. As a final contribution, we evaluate the security of the improved protocol and study the performance overhead requirements to achieve better security.

The rest of this paper is organized as follows. In Section 2, we cover the related work and give an overview of current research. Section 3 presents the details of GAMTP. This is followed by the cryptanalysis of the protocol and the full-disclosure attack in Section 4. We present an enhanced version of the protocol (E-GAMTP) in Section 5. The security and performance analysis of the modified protocol are presented in Sections 6 and 7, respectively. The paper is concluded in Section 8.

2. Related Work

Several grouping proof protocols have been proposed in order to meet the requirements of the providing grouping proofs within RFID systems.

In 2004, Jules presented the concept of the yoking proof [5] to give evidence that two tags have been simultaneously scanned. Jules proposed two protocols: the first utilizes message authentication code (MACs) and the second is based on a minimalist MAC scheme. Saito and Sakurai [18] proved that the protocols in [5] are not secure against replay attack and presented a scheme to solve the problem using timestamps. They further extended Jules' yoking proof to a grouping proof protocol. However, Piramuthu [19] showed that the Saito and Sakurai's protocol is not immune to reply attack and presented a modified proof that ensures dependence between tags so the reader cannot generate the proof without the simultaneous presence of both tags.

The chaining proof protocol was proposed by Lin et al. [20]. The main point in this protocol is to combine the target tags into a chain for integrity preservation. Similar to its predecessors, the protocol has two major shortcomings: first, the identities and some secrets of the tags are transmitted in plaintext, which makes the tags vulnerable to tracing attacks. Secondly, the protocol requires that the tags must be arranged in a fixed sequence before generating the proofs which may not be feasible in practical situations. Bolotny and Robins [21] introduced the anonymous yoking proof and generalized Jules' protocol to scan a group of tags instead of pair of tags simultaneously. Another anonymous yoking proof called "clumping proofs" appeared in [22]. However, Lo and Yeh [23] found that clumping proofs are not resistant against Denial of Proof attack which aims to ruin the normal identification of authorized tags [24]. This attack succeeds if the attacker adds illegal tags to the object and includes them in the proof generation. The BPS, as a result, cannot verify their partial proofs because their identity pseudonyms do not exist in the database and, thus, no secret keys for them will be found. Accordingly, the BPS will reject the groping proof. In addition, Lo and Yeh pointed out that clumping proofs lack forward security and do not prevent race conditions from happening.

Burmester et al. [25] introduced the group secret key to be used by all tags that belong to the same group. The authors presented three grouping protocols; the first is an offline grouping proof without anonymity, the second adds anonymity to the proof, and the third protocol is an online grouping proof designed with forwarding secrecy. These protocols were shown to be vulnerable to multiple impersonation attacks [6].

Further grouping proof protocols were analyzed in [6] to evaluate whether they meet their promised security goals or not. A traceability attack on the protocol in [26] was shown to be possible. The protocols in [27, 28], which are targeted towards enhanced inpatient medication safety, were shown

to fail to detect forged proofs. To supplement the work, the authors provided guidelines to be considered when designing grouping proof protocols. These include computing capabilities, matching, forward security, identification, verification, performance, and dependency. Based on these guidelines, a yoking protocol called Kazahaya was presented.

Bagheri et al. [29] applied passive full disclosure to obtain all the secrets with a cost of $O(2^{16})$ offline PRNG evaluations with a success probability of 1. They also presented a forgery attack that proves that a proof generated at time *tn* can be used to forge a valid proof at any following instance of time. Their work includes a modified protocol based on 128 bit PRNG to solve the problems of Kazahaya.

As a different approach to improve Kazahaya, the authors of [7] devised an offline yoking proof protocol using trusted timestamps. These are generated by a special tag, clock tag, in order to specify to the BPS the exact time at which the proof was generated.

Chen and Wu [30] proposed a novel yoking proof that conforming to the EPC global C1G2 standard with forward security. Their protocol is claimed to resist known attacks such as replay, man-in-the-middle, and tag tracking attacks.

Rostampour et al. [31] proposed a scalable grouping proof based on the use of a 64-bit pseudorandom number generator (PRNG). The protocol is considered secure against replay, impersonation, and traceability attacks. Furthermore, the protocol is shown to support partial proofs in which a subset of the tags can be incorporated in generating the proof. The authors provided a detailed BAN logic analysis to prove the message exchanges do satisfy the condition for which the reader and the tags can guarantee that no attacker has interfered in the message exchanges.

Lien et al. [8] first introduced the idea of reading-order-independent grouping proofs. In their scheme, the tags are read in an arbitrary order. The final proof is calculated by a pallet tag by applying the exclusive OR (XOR) operation to all intermediate values given by the tags. The pallet tag forwards the result to the BPS through the reader.

Although the protocol is order-independent, the tags cannot start the computation simultaneously since they have to wait for a partial value from the reader to be able to proceed [32]. In addition, the protocol is not resistant to desynchronization attack as mentioned in [9].

A grouping proof protocol conforming to the EPC C1-G2 standard with forward security was suggested [10]. The main operations used in the protocol are XOR and a 128-bit pseudorandom number generation. The protocol's shortcomings were shown in [9]. These include the lack of resistance to desynchronization and invalid proof generation attacks. From a practical point of view, the storage requirements are high and the necessity for using secure and trusted timestamp servers to generate the proofs render the protocol impractical.

The aforementioned protocols considered the case of a single reader accessing multiple tags. As a further step, the work in [24] considers the case of multireader distributed RFID systems. Three cases are covered: multitag with single reader, multireader with single-tag, and multitag with multiple-reader. The protocol is not immune to desynchronization attack and privacy attack as shown in [9].

Dhal and Gupta [33] proposed a lightweight authentication protocol that assumes that the large object has multiple tags attached to it. If any of these tags is identified, then the object is verified successfully. Based on this concept, Shen et al. [17] proposed a new offline lightweight RFID grouping authentication protocol for multiple tags suitable for large object identification. The proposed protocol only uses simple bitwise operation such as XOR, addition, and subtraction in order to comply with the capabilities of the passive tags.

In this work, we study the details of the protocol in [17] and show in detail how to apply a full-disclosure attack against it and then present modifications to improve its security levels.

3. Overview of GAMTP

In RFID grouping proof protocols, a tag is attached to the item to be identified. In the case of large items, such an approach might be ineffective as the tag could possibly reside out of the reading range of the reader.

To counter this effect, Shen et al. [17] proposed the GAMTP grouping proof protocol for identifying large objects, with increased reading reliability, by attaching multiple RFID tags. A subset of these tags is sufficient for proper item identification and authentication. The main idea behind the protocol is to create a grouping proof using the responses of the available tags only rather than waiting for all tags responses.

The protocol employs lightweight operations like the XOR, OR, addition, and subtraction. However, their use for cryptographic purposes renders the whole protocol insecure to various attacks against the system.

GAMTP is executed in four phases: initialization, tag acquisition, main authentication, and verification. In GAMTP, it is assumed that n tags $(T_1, T_2, \ldots T_n)$ belonging to a unique group, G, are attached to a large object. The communication between the BPS and the reader is assumed to be secure. Table 1 shows the notation used in GAMTP.

By considering that $[M]_k$ denotes the k^{th} bit of a string M, where $M \in \{r_{R_m}, ID_{R_m}, GID_g, r_{T_i}, e_i, S_g, r_{e_i}, M_g, S_{T_i}, Z_{GID_g}\}$, the message M is L-bit long, where the index of the LSB is 0 and the index of MSB is L-1. Accordingly, M is represented as

$$M = M_{(L-1)}, M_{(L-2)}, \ldots, M_{(1)}, M_{(0)} \quad (1)$$

The protocol is executed according to the following phases.

Initialization Phase. In this phase, all tags in group G are initialized with the group information and a sequence number, e_i, for all $i = 1, 2, \ldots, n$. Group information contains a unique identifier GID_g and a secret group key S_g.

Tag Acquisition Phase. The reader interrogates the tags then registers those that respond with verified information as *Available* (based on their stored values in phase 1). Available tags are responsible for generating the grouping proof in the next phase (main authentication phase) (note that the next phase may or may not immediately succeed tag acquisition

Table 1: Notation of the SGP protocol [17].

Symbol	Description
R_m	The m^{th} reader
G	A tag group
T_i	The i-th tag in group G
n	The number of tags in G
r_{R_m},r_{e_i}	Pseudo-random numbers generated by the BPS
ID_{R_m}	The identifier of R_m.
GID_g	The group identifier of G
r_{T_i}	The pseudo-random number generated by T_i
e_i	The sequence number of T_i
S_g	The secret group key for G
S_{T_i}	The secret of T_i
Z_{GID_g}	The generated proof of G
$TempID_{T_i}$	The temporary identifier of T_i

phase). Note that not all tags in a given group are required to respond but rather the available tags are sufficient to prove that the object is authentic.

This phase runs as follows:

(1) The BPS generates a random number r_{R_m} and determines the group to be verified. It then sends r_{R_m} along with ID_{R_m} and GID_g to the reader R_m.

(2) R_m receives the request from BPS and computes the message M_g as

$$M_g = \left(GID_g \bigoplus ID_{R_m}\right) + \left(S_g \vee r_{R_m}\right) \quad (2)$$

and then broadcasts it along with ID_{R_m} and r_{R_m} to all the tags in the group G.

(3) The tag T_i receives the message M_g and checks its validity by computing M_g locally and comparing it with that received. If found equal, the tag considers the reader as trusted. Next, T_i generates its pseudorandom number r_{T_i} and computes two reply messages:

$$N_{T_i} = \left[M_g - \left(GID_g \bigoplus ID_{R_m}\right)\right] \vee r_{T_i} \quad (3)$$

$$Q_{T_i} = e_i \bigoplus \left(S_g \vee r_{T_i}\right) \quad (4)$$

The tag T_i sends these two messages along with r_{T_i} to the reader $r\ R_m$.

(4) The reader R_m verifies the received message N_{T_i} and then extracts e_i from Q_{T_i} according to the equation $e_i = Q_{T_i} \bigoplus (S_g \vee r_{T_i})$. Once the reader has collected the IDs of all tags in the group (or a subset of them), it securely sends all the values of e_i to the BPS together with ID_{R_m} and GID_g.

(5) The BPS receives the information from the reader R_m and compares it with the stored data for authentication. Next, it generates a random number r_{e_i} for each tag T_i in order to be used in generating the grouping proof in next phase. These values are sent to the reader as $(ID_{R_m}, GID_g, r_{e_1}, \ldots, r_{e_i})$.

The steps of the tag acquisition phase are illustrated in Figure 1. For all the figures in this work, the top boxes indicate the known values for each entity.

Main Authentication Phase. The grouping proof is generated in this phase by chaining the tags according to their sequence number, e_i. The reader starts with the first tag in the group and then proceeds on building a dependency between the received responses and those from the subsequent tags. The authentication process proceeds sequentially from one tag to the next until all tags have responded. In case of a nonresponding tag, the reader then waits for a timeout period has passed before proceeding to the next tag in sequence.

The steps executed in this phase are as follows.

(1) The reader R_m starts with the first tag, T_1, and computes the following equations for $i = 1$:

$$r'_{T_i} = r_{T_i} \bigoplus r_{e_i}, \quad (5)$$

$$TempID_{T_i} = \left(GID_g \bigoplus e_i\right) \vee \left(r'_{T_i} \bigoplus ID_{R_m}\right) \quad (6)$$

and then sends Q_{T_1} (from the previous phase), $TempID_{T_1}$, r_{e_1}, and the flag "First" to the first tag in the group G.

(2) When T_1 receives the message, it locally computes $TempID_{T_1}$ and compares it with received one. If $TempID_{T_1}$ is verified successfully, then T_1 computes two responses Z_{GID_g} and C_{T_1} to be sent to R_m, where

$$Z_{GID_g} = TempID_{T_1} \vee \left(r_{e_1} + S_{T_1}\right), \quad (7)$$

$$C_{T_1} = \left(e_1 \bigoplus r'_{T_1}\right) \vee N_{T_1} \quad (8)$$

(3) R_m computes C_{T_1} locally and then compares it with the received C_{T_1}. If C_{T_1} is verified successfully, then R_m proceeds sequentially to the next tag T_i (for all $i = 2, \ldots, n$) and computes (5) and (6) and then sends the messages Q_{T_i}, $TempID_{T_i}$, r_{e_i}, and Z_{GID_g} to T_i.

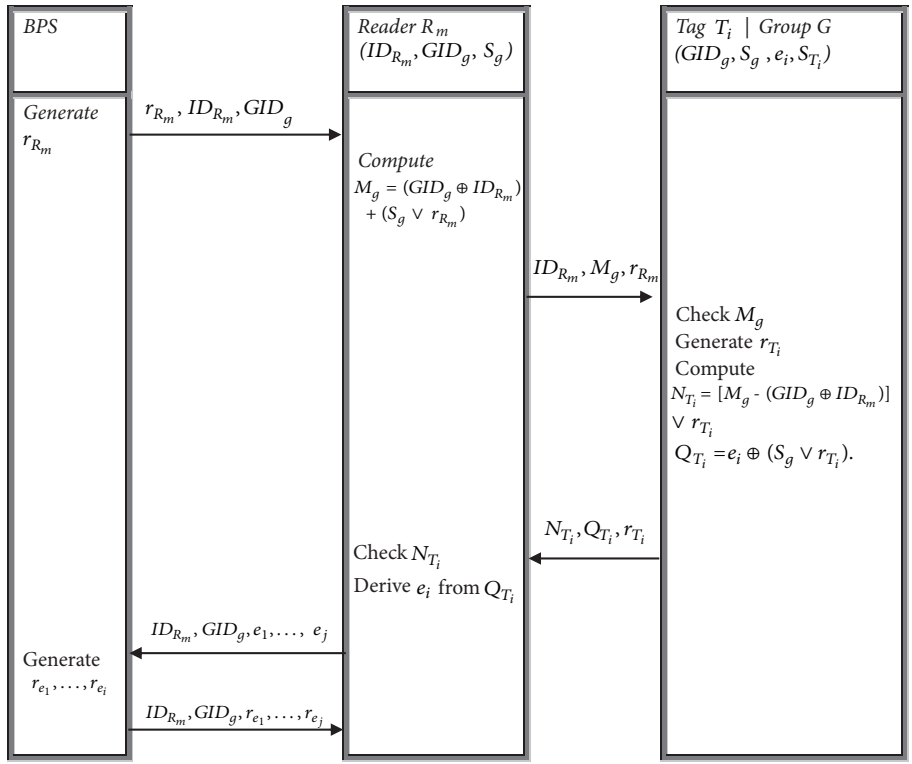

Figure 1: Tag acquisition phase of the GAMTP protocol [18].

Note that the message Z_{GID_g} that was computed by T_1 is sent to the next tag T_i in the group and G to ensure the dependency.

(4) The next tag in the sequence, T_i, checks the validity of $TempID_{T_i}$ and then computes

$$Z_{GID_g} = \left[TempID_{T_i} \vee \left(r_{e_i} + S_{T_i}\right)\right] \bigoplus Z_{GID_g} \quad (9)$$

$$C_{T_i} = \left(e_i \bigoplus r'_{T_i}\right) \vee N_{T_i} \quad (10)$$

Unlike in (7) which is used for the first tag only, each T_i updates Z_{GID_g} in (9) using the old Z_{GID_g} received from its predecessor tag. This is done in order to build the chain between the tags. The tag sends (Z_{GID_g}, C_{T_i}) to R_m.

(5) R_m repeats the same process with all the available tags of group G until it reaches the last tag through which the required grouping proof Z_{GID_g} is generated.

These steps are shown in Figure 2.

Verification Phase. As a final phase, R_m submits the proof Z_{GID_g} with all the sequence numbers of the available tags $(ID_{R_m}, GID_g, Z_{GID_g}, e_1, \ldots, e_j)$ to the BPS through a secure channel. In this way, the BPS will be informed of the tags that were not part of the proof generation.

4. Cryptanalysis of GAMTP

GAMTP is claimed to be secure against various attacks on RFID systems. However, we show in this work a three-step full-disclosure attack against it by exploiting the inherent weaknesses of the operations used for cryptographic purposes.

The operations used in the protocol are XOR, OR, addition, and subtraction operations. The use of OR operation to encrypt/construct response messages is a serious vulnerability since inference about its operands can be easily obtained (i.e., zero output of the OR operation reveals all the operands). Moreover, the addition operation does not hold any cryptographic strength, especially when some of the bits of the operands are known.

Upon completion of the disclosure attack, we expose all the secrets within the grouping proof protocol (S_g, GID_g, e_i, S_{T_i}).

The proposed attack is executed in three steps. Each step reveals partial or full string of a secret value. The output of step is used as an input for the subsequent step.

Step 1. The goal of this step is to get the group secret information: S_g, GID_g, and the sequence numbers of the tags in the group, e_i. The attacker eavesdrops on the exchanged messages between the reader and the tags during the tag acquisition phase until all the bits of the secrets are disclosed.

The attacker eavesdrops on the messages $(M_g, ID_{R_m}, r_{R_m}, N_{T_i}, Q_{T_i}, r_{T_i})$ that are exchanged between the reader and

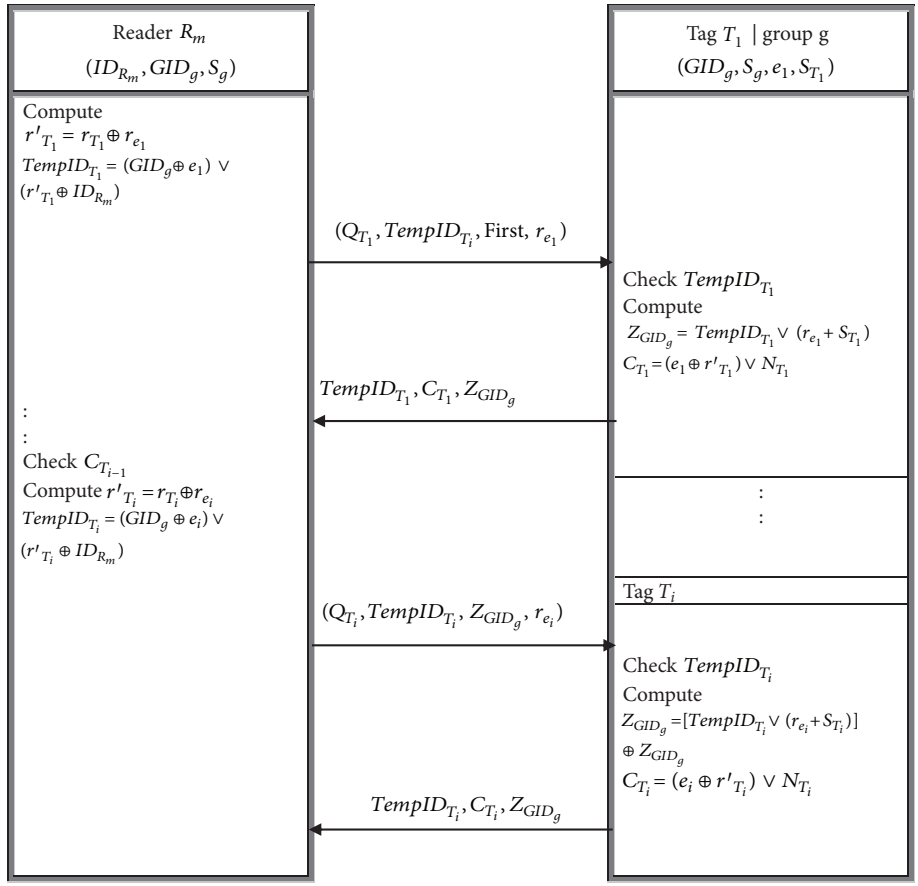

Figure 2: Main authentication phase of the GAMTP protocol [18].

the tags during the tag acquisition phase of a protocol session. In each session, the reader communicates with n tags and, as a result, the attacker gets the messages $(M_g, ID_{R_m}, r_{R_m}, N_{T_i}, Q_{T_i}, r_{T_i})$ for $T_1, T_2 \ldots T_n$. Note that the number of tags n differs from one session to another based on the available tags.

The attacker starts with the messages of the first tag T_1 and substitutes M_g into (3) to get

$$N_{T_1} = \left[\left(GID_g \oplus ID_{R_m}\right) + \left(S_g \vee r_{R_m}\right) - \left(GID_g \oplus ID_{R_m}\right)\right] \vee r_{T_1} \quad (11)$$

which is simplified to

$$N_{T_1} = S_g \vee r_{R_m} \vee r_{T_1} \quad (12)$$

If $([r_{R_m}]_k = [r_{T_1}]_k = 0)$, then $[S_g]_k = [N_{T_1}]_k$. Using this, an attacker can get the corresponding bits of the secret S_g for all k where the condition is met. If there are still some unknown bits in S_g, the attacker proceeds with the messages of the next tag within the group G until all bits S_g of are revealed.

With the secret value of S_g in hand, the attacker computes GID_g by substituting it into (2) along with the values that were collected during the same round $(M_g, r_{R_m}, ID_{R_m}, Q_{T_i}, r_{T_i})$. This results in

$$GID_g = \left(M_g - \left(S_g \vee r_{R_m}\right)\right) \oplus ID_{R_m} \quad (13)$$

The attacker also finds e_i, for all $i = 1, 2, \ldots, n$, by manipulating (4) and substituting the value of the revealed S_g to get.

$$e_i = Q_{T_i} \bigoplus \left(S_g \vee r_{T_i}\right) \quad (14)$$

Step 2. The next values to be revealed are the tag secrets S_{T_i} for all tags $i \in G$. Once S_{T_i} are obtained, the attacker can generate a fake proof to mislead the BPS to believe that a missing object is present. Even worse, it can change the secret values in the tags to compromise the whole grouping proof system. Here, we cover the case for revealing S_{T_j} for some tag $j \in G$. We emphasize the fact that this can be iterated for all other tags to find S_{T_i} for all tags $i \in G$.

First, during the main authentication phase, tag T_j calculates Z_{GID_g} as evidence to the BPS that it is participating in the generation of the proof, as given in (7). For this to be possible, T_j must be handled as the first tag in the chain using the "First" flag.

We observe that when $[r_{e_j}]_k = 0$ for all $k \in \{0, 1, \ldots, L-1\}$, then $Z_{GID_g} = TempID_{T_j} \vee S_{T_j}$. Furthermore, if $[TempID_{T_j}]_k = 0$ for all $k \in \{0, 1, \ldots, L-1\}$, then $[S_{T_j}]_k = [Z_{GID_g}]_k$.

Another observation is related to the computation of $TempID_{T_j}$ in (6). Let us denote the term $(GID_g \bigoplus e_j)$ as X. The value of the term X is constant through all the protocol sessions because GID_g and e_j are not updated at the end of any protocol session. Thus, if $[X]_k = 1$ then $[TempID_{T_j}]_k = 1$ for all $k \in \{0, 1, \ldots, L-1\}$ in all sessions.

On the other hand, if $[X]_k = 0$, then $[TempID_{T_j}]_k$ depends on the result of the term $(r'_{T_j} \oplus ID_{R_m})$ which is random. In this case the adversary needs to perform the following procedure.

(1) The attacker generates a random number and sends it to the target tag as r_{R_m}. Then, the attacker communicates with the target tag to execute the tag acquisition phase. This is possible because the adversary has gained possession of the needed secret values during the first step of the attack. When a response from the target tag is received, the attacker stores r_{T_j} for later use in the main authentication phase of the protocol.

(2) In the main authentication phase, the attacker sets $[r_{e_j}]_k = 0$ for all $k \in \{0, 1, \ldots, L-1\}$ and sends the flag "First" to inform the tag that it is the first tag in the chain of the main authentication phase. Next, the attacker computes $TempID_{T_j} = (GID_g \oplus e_j) \vee (r'_{T_j} \oplus ID_{R_m})$ and sends $(Q_{T_j}, TempID_{T_j}, \text{First}, r_{e_1} = 0)$ to the tag T_j.

(3) Tag T_j verifies the $TempID_{T_j}$ and computes $Z_{GID_g} = TempID_{T_j} \vee (r_{e_j} + S_{T_j})$, where $r_{e_1} = 0$, and $C_{T_j} = (e_j \bigoplus r'_{T_j}) \vee N_{T_j}$. The values $(TempID_{T_j}, C_{T_j}, Z_{GID_g})$ are sent to the attacker.

The attacker exploits the value of Z_{GID_g} by checking the bit positions for which $[TempID_{T_j}]_k = 0$ which means that $[S_{T_j}]_k = [Z_{GID_g}]_k$. Going through all the bits, the attacker will have S_{T_j} represented as

$$\left[S_{T_j}\right]_k = \begin{cases} \left[Z_{GID_g}\right]_k, & \text{for all } k \in \{0, 1, \ldots, L-1\} \text{ where } \left[TempID_{T_j}\right]_k = 0 \\ U, & \text{for all other } k \end{cases} \tag{15}$$

where U represents the value of a bit that is yet to be determined.

Capturing the messages of several sessions may be needed to cover all the bits at which $[TempID_{T_j}]_k = 0$.

Step 3. This step is needed to reveal the remaining bits of S_{T_j} that were not discovered in step two (i.e., the bits of S_{T_j} at the positions that correspond to those of the 1's in the term X). This is done by exploiting the attributes of the addition operation.

According to (7), if $[TempID_{T_j}]_k = 1$, then $[Z_{GID_g}]_k = 1$ regardless of the other values. Thus, there is no hint about the value of $[r_{e_j} + S_{T_j}]_k$ because it is masked by $[TempID_{T_j}]_k$. Fortunately, these masked bits may have neighboring bits with known values from the previous step from which we can derive further information.

To achieve this, the attacker must perform an active attack to get the undisclosed bits of S_{T_j} starting with the LSB up to the MSB. The main authentication phase is repeated several times with different values of r_{e_j} in order to mislead the tag to respond with a new r'_{T_j} random value.

To set the foundation for this step, let us assume that we have m unknown consecutive bits starting at bit position w (i.e., $[S_{T_j}]_k = U$ for $k \in \{w, w+1, \ldots, w+m-1\}$). Also, we consider that bits to the right w are known, $([S_{T_j}]_k = [Z_{GID_g}]_k$ for $k \in \{0, 1, \ldots, w-1\}$). We see that $[TempID_{T_j}]_{w-1} = [TempID_{T_j}]_{w+m} = 0$ because $[S_{T_j}]_k$ for $k \in \{0, 1, \ldots, w-1, w+m-1\}$ are known from step two.

Revisiting (7), we can deduce that

$$[C_{out}]_{w+m} = \left[r_{e_1}\right]_{w+m-1} \bigoplus \left[S_{T_j}\right]_{w+m-1} \bigoplus \left[Z_{GID_g}\right]_{w+m-1} \tag{16}$$

where $[C_{out}]_w$ is the resulting carry out from the addition of the bits at position w. This is because $[TempID_{T_j}]_{w+m} = 0$, which eliminates the OR operation from the equation. Based on (5) and (6), this also leads to

$$\left[r'_{T_j}\right]_{w+m} \bigoplus \left[ID_{R_m}\right]_{w+m} = \left[r_{T_j}\right]_{w+m} \bigoplus \left[r_{e_j}\right]_{w+m} \bigoplus \left[ID_{R_m}\right]_{w+m} = 0 \tag{17}$$

To exploit the addition operation in (7), we use the property stating that for $A + B$ (where A and B are binary strings and A is constant), the resulting carry out is equal to 0 ($C_{out} = 0$) iff the string B is less than or equal to the 1's complement of A. For example, if $A = 011$, $C_{out} = 0$ if and only if $B \in \{000, 001, 010, 011, 100\}$.

The attacker impersonates the reader to the target tag and starts by executing the tag acquisition phase to receive a new value of r_{T_j}. This is followed by the main authentication phase. Here, the attacker initially sets r_{e_j} as

Table 2: Average number of session needed to disclose S_g in Step 1 of the attack.

String Size (Bits) or $\lvert S_g \rvert$	Average number of sessions for 1000 trials
16	12.002
32	14.879
48	15.821
64	17.117
80	17.447
96	18.577

$$\left[r_{e_j}\right]_k = \begin{cases} 0, & \text{for } 0 \le k \le w-1 \text{ and } \mathrm{w}+\mathrm{m}+1 \le k \le L-1 \\ 1, & \text{for } \mathrm{w} \le k \le w+m-1 \\ \left[r_{T_j}\right]_k \bigoplus \left[ID_{R_m}\right]_k, & k = w+m \end{cases} \tag{18}$$

and then sends $(Q_{T_j}, TempID_{T_j}, \text{“First”}, r_{e_j})$ to the target tag. Note that the target tag is treated as the first tag in order to enforce using (7) rather than (9).

When the tag responds with Z_{GID_g}, the attacker checks the value of $[C_{out}]_{w+m}$. If it is equal to 1, the value of the bit string $[r_{e_j}]_k$ for $\mathrm{w} \le k \le w+m-1$ must be decremented by 1 and the step repeated again. If $[C_{out}]_{w+m} = 0$, then we consider that the r_{e_j} value that was sent to the target tag is correct. Thus, the missing bits of $[S_{T_j}]_k = \overline{[r_{e_j}]_k}$ for $k \in \{w, w+1, \ldots, w+m-1\}$ (i.e., the 1's complement).

The newly found bit of S_{T_j} can be incorporated within the string to repeat the same procedure and find more bits to the left of $w+m-1$.

For evaluation, we executed Step 1 (attack to discover all bits of S_g) of the proposed full-disclosure attack with different string sizes. The attack was executed in 1000 trials for each string size. In each trial, different random numbers are used in generating the grouping proof, namely, r_{R_m} and r_{T_i}. Table 2 depicts the average number of sessions needed to discover all the bits of S_g.

In Table 2, the logarithmic relation between the number of sessions and string size is evident. For instance, doubling the number of bits in S_g is reflected in 2 to 3 extra sessions to discover the bits. Such logarithmic relation signifies a serious weakness in the algorithm since the encryption is not tied to exponential, polynomial, or even linear complexity. Therefore, an increase in the key size (secrets and random numbers) will have a minor effect on the overall complexity of the attack.

To explain the results in Table 2, we note that the success in identifying any given k^{th} bit in the string S_g depends on having both and $[r_{R_m}]_k$ to be zero. In addition, since the RNG that is used to generate r_{T_i} and r_{R_m} is based on independent and identically distributed uniform function, $P([r_{T_i}]_k = [r_{R_m}]_k = 0) = P_{0,0} = 0.25$.

Thus, we can consider the probability of finding the value of a given bit in S_g after n sessions as

$$P_n = 1 - \left(1 - P_{0,0}\right)^n \tag{19}$$

where n is the number of executed sessions. Consequently, the expected number of known bits E_{S_g} can be expressed as $E_{S_g} = P_n \times |S_g|$. In our algorithm, we consider n that makes $\lceil E_{S_g} \rceil = |S_g|$ to be the number of sessions required to identify all bits in S_g.

Due to the logarithmic behavior of P_n, increasing the size of S_g by 25% will be reflected in 1 extra session; hence, the complexity of the attack will be $O(\log(|S_g|))$.

5. Enhanced GAMTP (E-GAMTP)

In this section, we propose an enhancement to the GAMTP (referred to as E-GAMTP) to overcome the vulnerabilities shown in the previous section. This enhancement is based on employing the conversion operation [34]. This is a stronger cryptographic bitwise function to resist the attack while, at the same time, avoiding addition of a high level of computational and power expensive operations.

The conversion has several important characteristics which are irreversibility, sensibility, full confusion, and low complexity. These features make it a suitable option to encrypt the secrets so that they cannot be exposed. The conversion operation accepts two n-bit strings as the input and consists of three steps:

(1) Grouping: A and B are divided into small blocks depending on their Hamming weights (hw) and a predefined threshold T.

(2) Rearrange: the blocks from the Grouping step are exchanged between A and B. Next, each block is left-rotated for a number of times equal to the block's Hamming weight. A' and B' will now hold the rotated blocks.

(3) Composition: the final result of the operation is produced by Xoring the strings from the Rearrange step ($Con(A, B) = A' \bigoplus B'$). An example of a

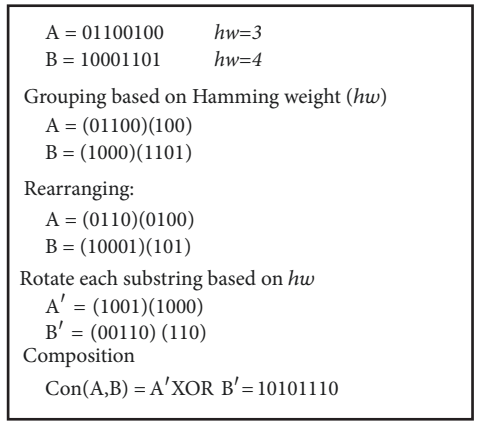

Figure 3: Example of a conversion operation between two 8-bit strings.

conversion operation for two 8-bit strings is given in Figure 3.

One of the main advantages of using the conversion operation is that it is irreversible. This stems from the fact that the input space is larger than the output space. As a result, the same result of the conversion operation may be produced for numerous different inputs. We make use of this property by using it as a one-way hash function for which a digest of the inputs is produced. The receiver also uses the operation in the forward direction by knowing the inputs and producing a hash to be compared with that received.

On another front, the original protocol had a problem related to message replay, through which an attacker could track the tags by sending old messages and waiting to receive the response of the targeted tags. Accordingly, the location and identity privacy of the tags would be compromised. To overcome this issue, we propose the use of timestamps for the exchanged messages to give the tag the capability to discern whether a received message is a fresh or an old (replayed) message. In the case of the latter, the tag simply does not respond to the claimed reader.

On a final note regarding the proposed enhancements, it is not feasible to use key updates or tag identity up.

E-GAMTP maintains the structure of the GAMTP as follows:

Initialization phase is the same as the original protocol.

In *tag acquisition phase*, we modify the messages of this phase by using the conversion operation. The general message structure remains the same.

Message M_g, as computed in (2), is modified to

$$M_g = Con\left(\left(GID_g \bigoplus ID_{R_m} \bigoplus r_{R_m}\right), \left(S_g \bigoplus r_{R_m} \oplus T_s\right)\right) \tag{20}$$

Another point to be addressed is related to the fact that the values of ID_{R_m} and r_{R_m} were broadcast as plaintext in the original protocol. Here, we propose to hide these values by encrypting them as

$$A = r_{R_m} \bigoplus Con\left(GID_g, S_g\right) \tag{21}$$

$$B = ID_{R_m} \bigoplus GID_g \tag{22}$$

$$C = T_s \bigoplus Con\left(r_{R_m}, S_g\right) \tag{23}$$

Then the reader broadcasts (A, B, C, M_g) messages to the tags. When the tag receives the messages, it verifies the reader and then extracts T_s from message C. If the value of T_s does not reflect a new message with a higher timestamp then the tag would not respond back. Otherwise, the new value of T_s is stored in the tag's memory as the most recent timestamp. This way, old messages become useless after being used for the first time.

Note that step one of the attack exploited the plaintext transmission of the random number r_{T_i} and the weak construction of messages N_{T_i} and Q_{T_i}. These weaknesses are avoided by encrypting r_{T_i} using the secret value S_g,

$$D = r_{T_i} \bigoplus r_{R_m} \bigoplus S_g, \tag{24}$$

and modifying the operations used to compute N_{T_i} and Q_{T_i} as

$$N_{T_i} = Con\left(\left(S_g \bigoplus r_{T_i} \bigoplus T_s\right), \left(ID_{R_m} \bigoplus GID_g \bigoplus r_{T_i}\right)\right) \bigoplus M_g \tag{25}$$

$$Q_{T_i} = e_i \bigoplus Con\left(\left(S_g \bigoplus r_{T_i}\right), r_{T_i}\right) \tag{26}$$

Note that the computation of message N_{T_i} involves the use of T_s to give the reader further guarantees that the messages are not replayed from previous sessions.

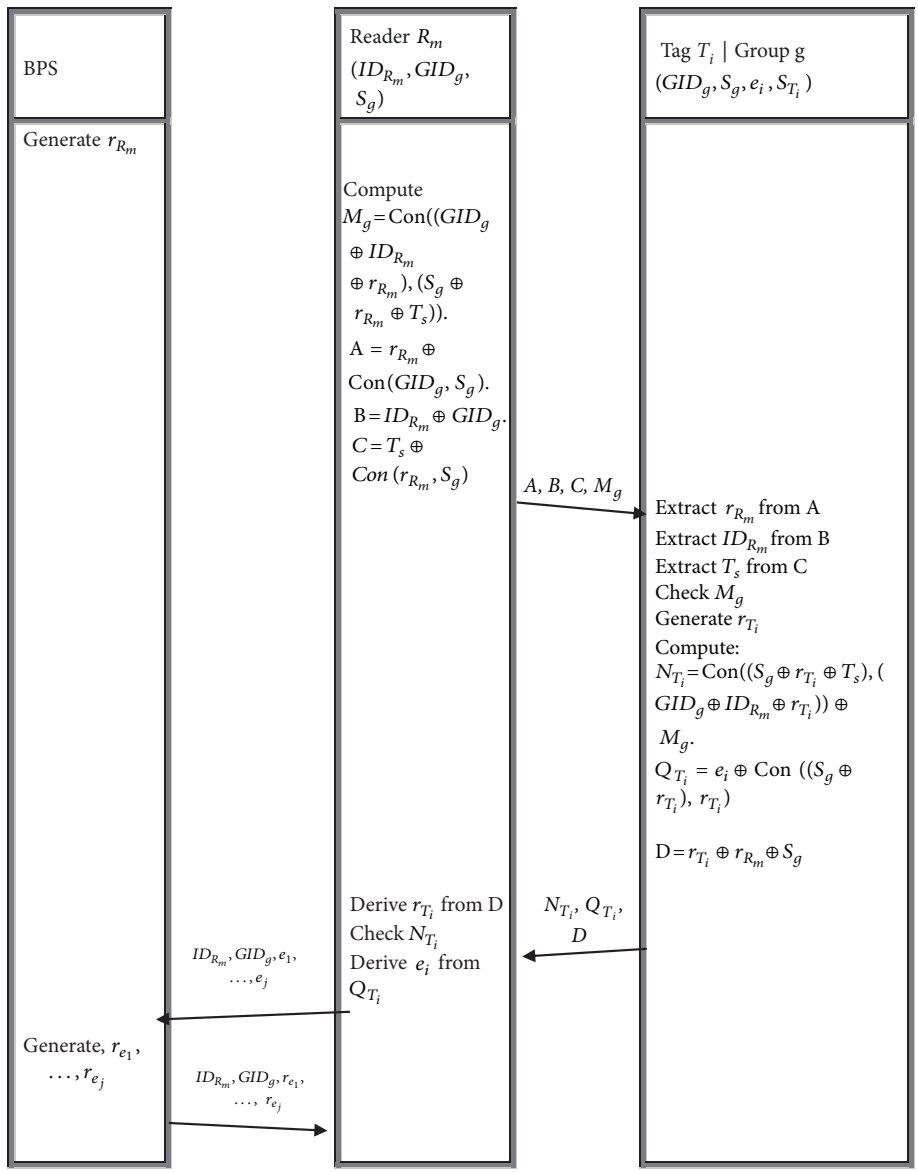

Figure 4: Tag acquisition phase of E-GAMTP protocol.

Main Authentication Phase. This phase is modified such that the values of r_{e_i}, $TempID_{T_1}$, Z_{GID_g}, and C_{T_i} are computed in a more secure manner.

$$E = r_{e_i} \bigoplus r_{T_i} \bigoplus S_g \tag{27}$$

$$TempID_{T_1} = Con\left(\left(r_{e_i} \bigoplus e_i\right), ID_{R_m}\right) \tag{28}$$

$$Z_{GID_g} = Con\left(\left(r_{e_i} \bigoplus S_{T_i}\right), \left(r_{e_i} \bigoplus S_g\right)\right) \bigoplus Z_{GID_g} \tag{29}$$

$$C_{T_i} = Con\left(\left(e_i \bigoplus r_{T_i}\right), N_{T_i}\right) \tag{30}$$

The tag acquisition and main authentication phases are illustrated in Figures 4 and 5.

The Verification Phase. This phase runs as specified in GAMTP.

6. Security Analysis of the Improved Protocol

In this section, we evaluate E-GAMTP's resistance to several well-known attacks against RFID systems. Depending on the adversary model in [18], we assume the adversary has the ability to

(1) eavesdrop on all the messages over the channel;

(2) send messages to any tag T_i and to the reader R_m in the group G as a legitimate device;

(3) intercept, modify and, resend the messages;

(4) perform any of the lightweight operations used in implementing the protocol.

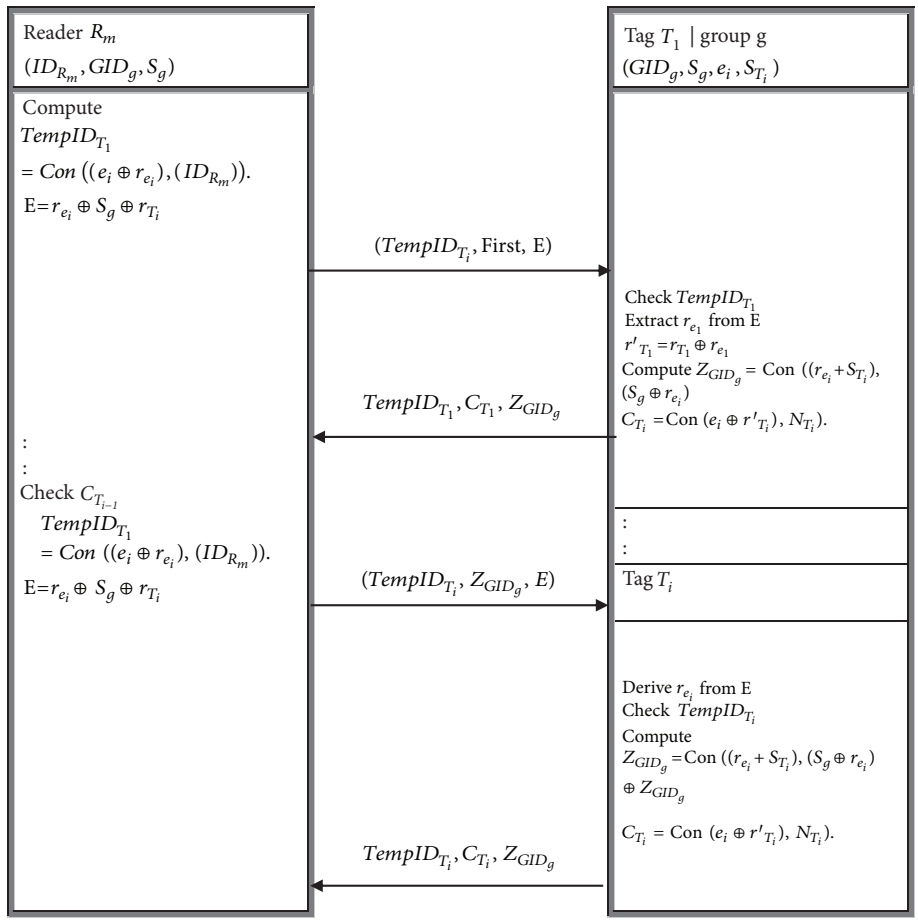

Figure 5: Main authentication phase of the E-GAMTP protocol.

Replay Attack. To implement the reply attack, the attacker needs to capture and save messages from a successful authentication session. When the reader requests the tags to participate in generating a proof, it broadcasts (A, B, C, M_g) messages to the tags in group G. When the tag detects that the timestamp of the message is old, it will not respond back to the reader.

Tag Impersonation Attack. The attacker performs the impersonation attack by convincing the reader to accept his messages as valid messages from a legitimate tag. By considering assumption 4 of the adversary model, the attacker blocks the tag's response to the reader (N_{T_i}, Q_{T_i}, D) and attempts to generate a new response.

For the attacker to succeed, he needs to generate a new message D based on a new random value $\hat{r}_{T_i}$ that would generate $\widehat{N}_{T_i} = N_{T_i}$ and $\widehat{Q}_{T_i} = Q_{T_i}$, for the same T_s. This translates to the third birthday problem (the birthday paradox). In such a problem, the user needs to generate two messages $D_1 \neq D_2$ for which (1) $\widehat{N}_{T_i} = N_{T_i}$ and (2) $\widehat{Q}_{T_i} = Q_{T_i}$. The probability of success for each goal is proportional to $2^{n/2}$, where n is the number of bits comprising the digest. This is represented as

$$\begin{aligned} \Pr\left(\widehat{N}_{T_i} = N_{T_i}\right) &= 1 - e^{-k(k-1)/2N} \\ \Pr\left(\widehat{Q}_{T_i} = Q_{T_i}\right) &= 1 - e^{-k(k-1)/2N} \end{aligned} \tag{31}$$

where k is the number of attempts needed to find the match and $N = 2^n$. Note that these are independent events and the probability of success becomes the result of their multiplication

$$\Pr(\textit{collision success}) = P = \left(1 - e^{-k(k-1)/2N}\right)^2 \tag{32}$$

To find the number of needed attempts with respect to P, rearrange (32) as

$$k = \sqrt{\ln \frac{1}{\left(1 - \sqrt{P}\right)} * 2N} \tag{33}$$

For EPC-C1G2 compliant tags, $n = 96$ bits. Thus for $P = 0.5$, $k \approx 441 \times 10^{12}$ attempts. If the duration taken before refreshing the random number and the timestamp is shorter than the time needed to execute these steps, then the attacker will need to start all over again.

Tracking Attack. The attacker is able to track a tag if there is a message which is constructed with constant values for every

Table 3: Security comparison result.

	a	b	c	d	e
Grouping proof [18]	N	P	N	N	P
Chaining proof protocol [20]	Y	P	N	N	N
GUPA [24]	Y	Y	Y	N	Y
APCMA [33]	Y	Y	Y	N	Y
GAMTP [17]	N	N	N	N	Y
E-GAMTP (this work)	Y	Y	Y	Y	Y

a: Replay, b: Eavesdropping, c: Tracking, d: Message modification, e:.Desynchronization.
Y: resistant, N: not resistant, and P: partially resistant.

session. In E-GAMTP, the tag uses fresh random numbers in computing N_{T_i} in (24), (25), and (26) in the tag acquisition phase. The same applies to the main authentication phase in the computation of Z_{GID_g} with a fresh random number, r_{e_i}, which is generated by the BPS. Thus, the messages originating from the tag will be fresh and the attacker cannot recognize any constant values in them. Finally, the incorporated timestamps guarantee that the tags would not respond to replayed messages.

Desynchronization Attack. Since no group secret values are updated at the end of any session, the attacker cannot break the synchronization between the tags and the reader.

Message Modification Attack. Based on our analysis with regard to the tag impersonation attack, the number of attempts needed by the attacker for a successful message modification will be very high to the extent that the attack would be infeasible within the duration between two consecutive protocol sessions. Accordingly, the proposed protocol is resistant to any modification attack.

To conclude this section, a summary of the achieved resistance against attacks is given in Table 3. Part of this table first appeared in [17] and is revised it by adding our findings regarding GAMTP. We further show that the proposed protocol addresses all the well-known attacks on RFID systems.

7. Performance Analysis

In this section, we discuss the performance of the E-GAMTP in comparison with other lightweight protocols designed for achieving grouping proofs.

Table 4 gives a summary of the comparison in terms of storage requirements (SR), communication overhead (CO), and computational load (CL). The protocols to be compared appeared in [6, 17, 23–25]. These values are based on the assumption of having two tags T_a and T_b and one reader.

With regard to the SR, a tag stores $GID_g, S_g, e_i, S_{T_i}, T_s$ and the reader stores $ID_{R_m}, GID_g, S_g, T_s$. Compared with GAMTP, only one value is added to the list, which is the timestamp. In general, E-GAMTP needs less storage compared to the other protocols except for [6, 23].

As for the CO, the number of exchanged data packets is 24 per session, which is relatively higher than most protocols except for GAMTP. Tags (T_a, T_b) transmit the packets to the reader in four steps and the reader four packets for both tags. Finally, E-GAMTP requires six rounds per tag to complete generating the grouping proof.

In terms of the CL, T_a performs 46 bitwise operations and 1 PRNG function, while T_b needs one additional XOR operation to complete the chaining part in the computation of Z_{GID_g}. The reader, on the other hand, executes 69 bitwise operations and two PRNG functions.

Compared with GAMTP, E-GAMTP runs the conversion operation which consists of three bitwise operations (two rotations and one XOR). However, the tradeoff between adding more security to the system at the expense of added computations is well-justified.

Other protocols use functions with higher CL, such as the hash and MAC functions in [23, 33]. Kazahaya has the least CL amongst these protocols given that the PRNG and bitwise operations are less demanding than the hash or MAC functions.

8. Conclusion

Grouping proof is an important functionality provided by an RFID system. Although various schemes were presented in the literature, the quest for finding a secure and efficient protocol has not succeeded as most of these protocols are vulnerable to one attack.

In this work, we evaluated the GAMTP and showed how it is possible to execute a full-disclosure attack to reveal all of the secret values with a success probability of one. Our cryptanalysis showed that the weaknesses of the protocol stemmed from the weak construction of the messages, as well as the lack of cryptographic efficiency of applying primitive bitwise operations. Furthermore, we showed that increasing the length of the strings by 25% will be reflected in 1 extra session of eavesdropping and that the complexity of the attack is $\boldsymbol{O}(\log(|\boldsymbol{S_g}|))$.

To overcome these weaknesses, we proposed an enhanced version, E-GAMTP. This protocol uses a similar structure to the original but employs the conversion operation, which results in higher cryptographic efficiency. A detailed security analysis of the enhanced protocol was given to show its resistance to various attacks.

Finally, we provided a detailed performance analysis in terms of SR, CO, and CL. Although there is no additional

Table 4: Performance comparison of E-GAMTP with other protocols.

		Burmester P3 [25]	Kazahaya [6]	OTSBP [23]	GUPA:2T-R [24]	GAMTP	E-GAMTP
SR	T	$6l$	$4l$	$3l$	$(3+2z')\,l$	$4\,l$	$4\,l$
	R(DB)	$(2+y)l$	$1l$	$1l$	$(3+2z+y)l$	$3\,l$	$3\,l$
CO	$T \longrightarrow R$(DB)	4	3	3	5	4	4
	R(DB)$\longrightarrow T$	5	3	3	4	4	4
	Total Packet	14	15	17	19	26	24
	Round Number	6	6	8	8	6	6
CL	T_a	2R+3H	13B+13R	4B+4R+2M	19B+R	14B+1R	46B+1R
	T_b	2R+3H	7B+10R	4B+4R+2M	19B+R	15B+1R	47B+1R
	R_i(DB)	1R	12B+13R	1H	30B+3R	27B+2R	69B+2R

l: length of identifier; x: the reader number; y: the tag number; z': the reader group number; z: the tag group number.
B: bitwise function; R: PRNG function; M: MAC function; H: hash function; E: encryption;

storage for E-GAMTP compared to the other protocols except for [6, 23], we witnessed an increase in the CO and CL. These added loads are well-justified as they result in better overall security of the system.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This research was conducted at Jordan University of Science and Technology.

References

[1] Y. Tian, G. Chen, and J. Li, "A new ultralightweight RFID authentication protocol with permutation," *IEEE Communications Letters*, vol. 16, no. 5, pp. 702–705, 2012.

[2] A. Juels, "RFID security and privacy: a research survey," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 381–394, 2006.

[3] G. Cong, Z. Zhang, L. Zhu, Y. Tan, and Y. Zhen, "A novel secure group RFID authentication protocol," *The Journal of China Universities of Posts and Telecommunications*, vol. 21, pp. 94–103, 2014.

[4] H. Li, Y. Chen, and Z. He, "The survey of RFID attacks and defenses," in *Proceedings of the 2012 8th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM)*, pp. 1–4, 2012.

[5] A. Juels, "Yoking-proofs for RFID tags," in *Proceedings of the 2nd IEEE Annual Conference on Pervasive Computing and Communications Workshops*, vol. 14, pp. 138–143, March 2004.

[6] P. Peris-Lopez, A. Orfila, J. C. Hernandez-Castro, and J. C. A. Van Der Lubbe, "Flaws on RFID grouping-proofs. Guidelines for future sound protocols," *Journal of Network and Computer Applications*, vol. 34, no. 3, pp. 833–845, 2011.

[7] C. Ma, J. Lin, Y. Wang, and M. Shang, "Offline RFID grouping proofs with trusted timestamps," in *Proceedings of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, TrustCom-2012*, pp. 674–681, June 2012.

[8] Y. Lien, X. Leng, K. Mayes, and J.-H. Chiu, "Reading order independent grouping proof for RFID tags," in *Proceedings of the IEEE International Conference on Intelligence and Security Informatics, 2008, IEEE ISI 2008*, pp. 128–136, 2008.

[9] B. Yuan and J. Liu, "A universally composable secure grouping-proof protocol for RFID tags," *Concurrency Computation*, vol. 28, no. 6, pp. 1872–1883, 2016.

[10] S. Sundaresan, R. Doss, S. Piramuthu, and W. Zhou, "A robust grouping proof protocol for RFID EPC C1G2 tags," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 6, pp. 961–975, 2014.

[11] M. Brsz, B. Boros, P. Ligeti, K. Lja, and D. Nagy, "Breaking LMAP," in *Printed handout of Workshop on RFID Security (RFIDSec 07)*, 2007.

[12] H. Chien and C. Huang, "Security of ultra-lightweight RFID authentication protocols and its improvements," *ACM SIGOPS Operating Systems Review*, vol. 41, no. 4, pp. 83–86, 2007.

[13] G. Avoine, X. Carpent, and B. Martin, "Privacy-friendly synchronized ultralightweight authentication protocols in the storm," *Journal of Network and Computer Applications*, vol. 35, no. 2, pp. 826–843, 2012.

[14] T. Li and R. Deng, "Vulnerability analysis of EMAP-An Efficient RFID mutual authentication protocol," in *Proceedings of the Second International Conference on Availability, Reliability and Security*, pp. 238–245, IEEE, 2007.

[15] D. Tagra, M. Rahman, and S. Sampalli, "Technique for preventing DoS attacks on RFID systems," in *Proceedings of the 18th International Conference on Software, Telecommunications and Computer Networks, SoftCOM 2010*, pp. 6–10, September 2010.

[16] H.-Y. Chien, "SASI: a new ultralightweight RFID authentication protocol providing strong authentication and strong integrity," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, no. 4, pp. 337–340, 2007.

[17] J. Shen, H. Tan, Y. Zhang, X. Sun, and Y. Xiang, "A new lightweight RFID grouping authentication protocol for multiple tags in mobile environment," *Multimedia Tools and Applications*, vol. 76, no. 21, pp. 22761–22783, 2017.

[18] J. Saito and K. Sakurai, "Grouping proof for RFID tags," in *Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA '05)*, vol. 2, pp. 621–624, IEEE, March 2005.

[19] S. Piramuthu, "On existence proofs for multiple RFID tags," in *Proceedings of the International Conference on Pervasive Services (ICPS '06)*, pp. 317–320, 2006.

[20] C. Lin, Y. Lai, J. Tygar, C. Yang, and C. Chiang, "Coexistence proof using chain of timestamps for multiple RFID tags," in *Advances in Web and Network Technologies, and Information Management*, pp. 634–643, 2007.

[21] L. Bolotny and G. Robins, "Generalized yoking-proofs for a group of RFID," in *Proceedings of theInternational Conference on Mobile and Ubiquitous Systems: Networking & Services*, pp. 1–4, July 2006.

[22] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. Estevez-Tapiador, and A. Ribagorda, "Solving the simultaneous scanning problem anonymously: clumping proofs for RFID tags," in *Proceedings of the 3rd International Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing (SecPerU '07)*, pp. 55–60, 2007.

[23] N. Lo and K. Yeh, "Anonymous coexistence proofs for RFID tags," *Journal of Information Science and Engineering*, vol. 26, pp. 1213–1230, 2010.

[24] H. Liu, H. Ning, Y. Zhang, D. He, Q. Xiong, and L. T. Yang, "Grouping-proofs-based authentication protocol for distributed RFID systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1321–1330, 2013.

[25] M. Burmester, B. de Medeiros, and R. Motta, "Provably secure grouping-proofs for RFID Tags," in *International Conference on Smart Card Research and Advanced Applications*, pp. 176–190, 2008.

[26] H. Chien and S. Liu, "Tree-based RFID yoking proof," in *Proceedings of the 2009 International Conference on Networks Security, Wireless Communications and Trusted Computing (NSWCTC)*, pp. 550–553, Wuhan, China, April 2009.

[27] H.-H. Huang and C.-Y. Ku, "A RFID grouping proof protocol for medication safety of inpatient," *Journal of Medical Systems*, vol. 33, no. 6, pp. 467–474, 2009.

[28] H.-Y. Chien, C.-C. Yang, T.-C. Wu, and C.-F. Lee, "Two RFID-based solutions to enhance inpatient medication safety," *Journal of Medical Systems*, vol. 35, no. 3, pp. 369–375, 2011.

[29] N. Bagheri, M. Safkhani, M. E. Namin, and S. Rostampour, "An improved low-cost yoking proof protocol based on Kazahaya's flaws," *The Journal of Supercomputing*, vol. 74, no. 5, pp. 1934–1948, 2018.

[30] C. Chen and C. Wu, "An RFID system yoking-proof protocol conforming to EPCglobal C1G2 standards," *Security and Communication Networks*, vol. 7, no. 12, pp. 2527–2541, 2014.

[31] S. Rostampour, N. Bagheri, M. Hosseinzadeh, and A. Khademzadeh, "A Scalable and lightweight grouping proof protocol for internet of things applications," *The Journal of Supercomputing*, vol. 74, no. 1, pp. 71–86, 2018.

[32] H. Sun, W. Ting, and S. Chang, "Offlined simultaneous grouping proof for RFID tags," in *Proceedings of the 2nd International Conference on Computer Science and Its Applications (CSA '09)*, pp. 1-2, 2009.

[33] S. Dhal and I. Gupta, "A new authentication protocol for RFID communication in multi-tag arrangement," in *Proceedings of the 2014 International Conference on Computing for Sustainable Global Development, INDIACom 2014*, pp. 668–673, India, March 2014.

[34] H. Luo, G. Wen, J. Su, and Z. Huang, "SLAP: Succinct and Lightweight Authentication Protocol for low-cost RFID system," *Wireless Networks*, vol. 24, no. 1, pp. 69–78, 2018.