An Anonymous RFID Grouping-Proof with Missing Tag Identification

Mike Burmester
Department of Computer Science
Florida State University, Tallahassee, FL, 32306, U.S.A.
Email: burmester@cs.fsu.edu

Jorge Munilla
Campus de Excelencia Internacional Andaluca Tech
Universidad de Malaga, Malaga, 29070, Spain
Email: munilla@ic.uma.es

***Abstract*—The emergence of RFID (Radio Frequency Identification) technology has greatly increased the efficiency for inventory control, supply-chain management and logistics. With RFID group scanning, an RFID reader scans a collection of RFID tagged objects to generate a grouping-proof of "simultaneous" presence. Some shipments may have to be tracked remotely by readers that are not necessarily trusted. Current solutions, particularly those supporting anonymity, require readers to be trusted. In this paper we study RFID group scanning applications with untrusted readers, and present an anonymous grouping-proof of integrity for collections of RFID tagged objects. The proof can be checked by the verifier (a trusted entity) and the reader (an untrusted entity) can recover the identifiers of missing tags, but cannot generate a proof if tags are missing. The protocol can easily be adapted to get an anonymous RFID grouping code and is very efficient with just two rounds. We only assume that tags are able to generate pseudo-random numbers and compute one-way hash functions.**

***Index Terms*—RFID grouping-proofs, grouping codes, anonymity, missing tag identification, forward error correction, erasure codes.**

I. Introduction

RFID (Radio-Frequency Identification) is an emerging wireless communication technology that has stimulated numerous innovative applications in several fields such as inventory control, supply-chain management and logistics, as well as identify new research challenges and opportunities. A typical RFID deployment has three main components: i) tags or transponders, which are electronic data storage devices attached to objects to be identified; ii) readers or interrogators, that manage tag population, read data from and write data to tags; and iii) a back-end server, the verifier, which is a trusted entity that exchanges tag information with the readers and processes data according to specific task applications. Most RFID tags are passive and do not have power of their own but get the energy needed to operate from an RFID reader. Passive tags are inactive until activated by the electromagnetic field generated by a reader tuned to their frequency.

Although initial designs of RFID systems focused on performance reliability with less attention paid to resilience and security, the technology has now found use in many applications that require the implementation of security mechanisms that: i) take into account features such as the vulnerability of the radio channel, the constrained power of devices, the low-cost and limited functionality of tags and the request-response operation mode; and ii) make them resistant to privacy/confidentiality threats, malicious traceability and loss of data integrity. The recently ratified EPC Gen2v2 standard confirms this interest in security [1].

When RFID technology is used for supply-chain management, concerns regarding the monitoring of tags and transfer of ownership or control of tags need to be addressed. If the transfer is permanent, or even temporal, ownership transfer protocols can be used [2], [3]. However there are cases when the owner does not want to cede control, even though this may be temporal. For example, a manufacturer may use services provided by a carrier who, in turn, uses other carriers to transport products. In such cases it is desirable that the owner can periodically check the integrity of a shipment via the carrier. This requirement is referred to as *group scanning*, and involves multiple tags generating a *grouping-proof* of "simultaneous" presence in the range of an RFID reader [4], [5].

There are several practical scenarios where grouping-proofs can substantially expand the capabilities of RFID-based systems. For example, some products may need to be shipped together and one may want to track their progress through the supply-chain—*e.g.*, hardware components of a system or kits. A different scenario would involve enforcement of safety regulations requiring that drugs be shipped, or dispensed, with information leaflets. More generally, grouping-proofs can be used to check the integrity of pallet shipments.

Our contribution: Despite considerable research interest, many of the proposed RFID grouping-proofs make assumptions that are not practical (e.g. RFID readers are trusted or tag singularization is omitted), and there are still some aspects that, as far as we know, have not been discussed. Most RFID protocols in the literature leak some privacy information. For example, the adversary can learn the *number* of tags that take part in the protocol and the *order* in which tags reply. With grouping-proofs these problems are related to tag-chaining in [6] where, each tag in the group authenticates a message coming from t a previous tag in the chain. In this paper, after defining clearly our design criteria, we address these two problems and describe a two-pass grouping-proof that allows an untrusted reader to identify missing tags, and if the group is complete, to compile a grouping-proof of integrity that the verifier can check. More specifically, we present an anonymous RFID grouping-proof of integrity for a collection of tagged

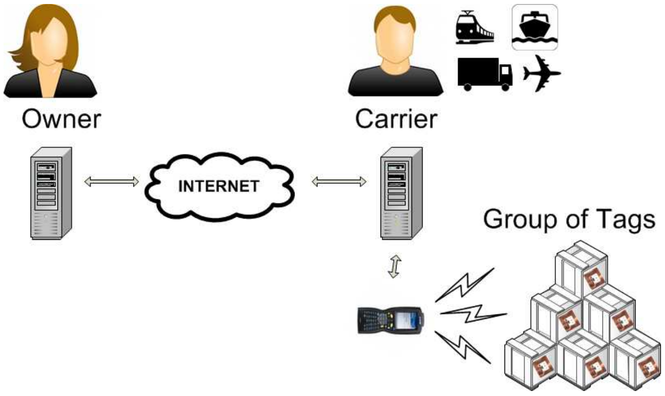

Fig. 1. An untrusted Carrier can identify any missing objects in a pallet and, when the group of tags is complete, compile a grouping-proof of integrity that the Owner can verify.

objects that supports tag privacy (in particular, untraceability) such that:

a) Only the verifier (a trusted entity) can check the proof.
b) The verifier can authorise an untrusted reader to inspect the group and identify any missing tagged objects.
c) The authorization is for one only inspection, and the tags are untraceable while the group is not inspected.
d) The reader cannot generate a grouping-proof for a group with missing tags.

The rest of this paper is organized as follows. In Section II we review the literature for RFID grouping-proofs and RFID grouping codes. In Section III we discuss RFID grouping-proof deployments, specifications, requirements and the threat model. In Section IV we discuss our objectives and present an anonymous RFID grouping-proof of integrity with missing tag identification. In Section V we discuss the security aspects. We conclude in Section VI by summarizing our main results.

II. Background

Ari Juels introduced in 2004 the security context of a new RFID application—which he called a yoking-proof [7], that generates evidence of simultaneous presence of two tags in the range of an RFID reader. The first protocol was later found to be insecure [8], [9] but, group scanning triggered considerable interest in the research community. Yoking-proofs were extended to grouping-proofs in which multiple tags prove simultaneous presence in the range of an RFID reader—see e.g. [10]. Burmester et al. presented in [11] a protocol that achieved anonymity by using randomized pseudonyms for the group identifer, and forward-security by updating the secret keys and the group keys after each session. This protocol is essentially a proof-of-concept, and not appropriate for lightweight applications. Huang and Ku [12] presented a grouping-proof for passive low-cost tags that uses a pseudorandom number generator to authenticate flows and a cyclic redundancy code to randomize strings. The protocol has several weaknesses, some of which were addressed by Chien et al. [13] who, in turn, proposed a new grouping-proof. Peris-Lopez et al. [6] found other security flaws in these protocols and proposed guidelines for securing them as well as a yoking-proof protocol (for two tags). More recently, Liu et al. [4] proposed a grouping-proof for distributed RFID applications with trusted readers. This proof is vulnerable to de-synchronization and privacy leaks [14].

While grouping-proofs provide integrity evidence for complete groups of tags, they do not address incomplete groups, in particular, they do not provide any information about missing tags. In 2012 Sato *et al.* [15] proposed a grouping code that makes it possible to find the identifiers of all tags of a group including missing tags, without requiring a packaging list or an external database. The code uses information previously encoded on each tag to determine if all the tags are present and if not, the identities of the missing tags. Such forward error correction mechanisms can increase the operating speed and reduce cost when it is difficult to access a database with the corresponding information. The Sato *et al*. grouping codes are based on Gallager low-density parity check (LDPC) codes [16]. However the randomised nature of Gallager LDPC codes makes it difficult to get specific decoding guarantees. To address this issue, Su *et al.* [17] proposed a LDPC variant that uses strongly selective families (SSF). Another approach, also based on SSF, is proposed in [18] to provide unequal protection to the tags. To improve on these codes, Su and Tonguz [19] proposed a variant that uses the Chinese remainder theorem (CRT) to construct non-regular generating matrices. This non-regularity makes it difficult to find general expressions for decoding guarantees. Another modification proposed by Su [20] uses resolvable transversal designs (RTD) to generate parity-check matrices and group splitting to improve performance. Finally Mabrouk and Couderc [21] propose an RFID grouping code that is based on Reed-Solomon (RS) codes. However the size of the blocks and the partitioning of the redundancy is not optimal.

III. Grouping-Proofs & Design Criteria

A. Group-scanning deployments

A typical deployment of an RFID grouping-proof involves three types of entities.

a) A group of tags G (GoT).
b) A verifying server or simply, *verifier*: the owner of G. The owner keeps the digital rights of the tags and knows the private information stored by the tags.
c) A *reader*: the carrier whose services are contracted by the owner. The carrier has physical possession of G and can access it through its reader(s), but does not control G.

A grouping-proof provides evidence of temporal events that corroborate the “simultaneous” presence of a GoT. A grouping proof is generated by the GoT if (completeness) and only if (soundness) all the tags of the group are simultaneously in the range of a reader (in practice, within the same interval window). It is important to note that when symmetric key cryptography is used, grouping-proofs are not real “proofs” in the sense that they are not transferable and can only be validated by those who share the private keys used to generate them.

B. Capabilities of group-scanning parties

Passive UHF tags are the most common for supply-chain applications. They have no power of their own, operate in the far field, and backscatter communication [22]. Such tags work at greater distances (than inductive tags) but the delivered power is low, and therefore lightweight cryptographic tools should be utilized [23]. However, we can assume that tags are able to perform basic symmetric-key cryptographic operations such as selecting pseudo-random numbers and evaluating a pseudo-random function. On-board clocks are beyond the capabilities of most tags, but the activity time span of a tag during a single session can be limited using techniques such as measuring the discharge rate of capacitors [7]. By contrast readers and verifiers/servers are able to perform complex cryptographic operations.

C. Erasure codes

Let $\mathbb{F}_q$ be a finite field of order q, $q = p^m$, p a prime, m a positive integer. A q-ary (n, k, s) erasure code is a linear forward error correction code that encodes source (input) data $x=(x_1, \dots, x_k) \in \mathbb{F}_q^k$ to encoded data $y=(y_1, \dots, y_n) \in \mathbb{F}_q^n$, in such a way that the source data can be recovered if no more than s blocks $y_i \in \mathbb{F}_q$ are missing. In a *systematic* code the source data is embedded in the encoded data. Typically, $y_i = x_i$, for $i = 1, \dots, k$. We must have $s \leq n-k$ (Singelton bound); the optimal case $s=(n-k)$ is reached with Maximum Distance Separable (MDS) codes. The most common MDS codes are the Reed-Solomon (RS) codes that are cyclic over $\mathbb{F}_q$, $q=p^m > n$, with minimum distance $d = n–k+1$ ($s=d–1$).

In our protocol in Section IV we shall use an $RS(n, k)$ code over $\mathbb{F}_{2^m}$, $2 \leq m \leq 16$ (according to RFC 6865 [24]), to encode the identifiers $(id_1, \dots, id_{n_g})$ of n_g RFID tags so that we can recover up to $s_t = (n-k)/(n/n_g)$ identifiers of missing tags. For this application the source data $x = id_1 \| \cdots \| id_{n_g}$ is an $n_g\ell$ bit string, where ℓ is the binary length of the identifiers id_i. We rearrange x into k blocks $x_i \in \mathbb{F}_{2^m}$ (depending on the implementation, some blocks x_i can be padded with zeros if necessary). Then x is encoded to get an RS codeword $(y_1, \dots, y_n)$, with n/n_g blocks stored in the memory of each tag_i, so that we can recover s_t missing tag identifiers. These n/n_g blocks stored in tag_i are denoted by ID_i and provide the identifying information provided by id_i as well as redundancy information that allows to recover the missing tags.

RS decoding can only be performed if the scanned identifiers y_i are ordered correctly, with gaps for the missing values. For this purpose control information is needed: the identifier ID_i of each tag_i is extended to include some extra bits that define its order i when it was encoded. As an example suppose that the $RS(150, 120)$ code over $GF(2^8)$ is used with a group of $n_g = 10$ tags. Then $k/n_g = 12\,\mathrm{bytes}$ are allocated for the tag identifiers ID_i (as required by the EPC Gen2v2 standard [1]), that are then extended to $n/n_g = 15\,\mathrm{bytes}$ to recover up to $s_t = (150-120)/15 = 2$ missing tags. In this case 4 bits are sufficient for control information. In total 124 bits are needed for the extended tag identifiers. For larger groups, say with $n_t = 100$ tags and up to $s_t = 60$ missing tags, we can use $RS(2000, 800)$ over $GF(2^{12})$: in this case the $k/n_g = 8\,\mathrm{symbols} = 12\,\mathrm{bytes}$ of the tag identifiers ID_i are extended to $n/n_g = 20\,\mathrm{symbols} = 30\,\mathrm{bytes}$, and 20 bits are sufficient for control information (4 bits for the value of m and 2 bytes for the value of n_g—up to 256 tags, and the order of the tag). In total an additional $15 \times 8 + 4 = 124\,\mathrm{bits}$ are needed to recover up to 60% of missing tags.[1]

Note that optimal codes are costly when n is large: encoding and decoding have quadratic complexity. However for our applications the number of tags and missing tags is typically small, and the computational complexity is born the RFID readers for which there are no computational or memory constraints, as opposed to RFID tags that are severely constrained in memory, communication and number of computations.

D. Threat Model for RFID systems

RFID wireless channels are particularly vulnerable because tags are restricted to lightweight cryptographic protection. We assume the Byzantine threat model in which the adversary controls the communication channels, and may eavesdrop, block, modify and or inject messages in any communication between tags and readers. In practice reader-tag (forward) channels are easier to intercept than tag-reader (backward) channels, since the signal in the latter case is much weaker. By contrast, the communication channel between high level entities (*i.e.* readers and verifiers) is secure since fully-fledged cryptographic techniques can be used. However, these channels may or may not enjoy continuous connectivity. Thus, during an interrogation the verifier may be online or offline, and different solutions for the grouping-proof problem are required in each case.

Several types of attacks against RFID systems have been described in the literature. Some are well known on other platforms. In particular, the adversary may attempt to perform impersonation, DoS, interleaving and reflection attacks and other passive or active attacks. The unique aspects of RFID applications highlight other vulnerabilities such as unauthorized tracking, a privacy concern in which the adversary tries to trace and/or recognize tags of a group. There are also attacks on RFID systems that are usually excluded from the security model used, such as *online man-in-the-middle relay attacks* [25] and *side channel* or *power analysis* [26] attacks. In particular, if no distance-bounding mechanism [27] is used, our protocol in Section IV will be subject to active attacks that involve relaying flows between tags faster than the time interval defined by the tag timers. These attacks affect all RFID protocols [28], including grouping-proofs [29], and can only be addressed by making certain that precise timing mechanisms are used.

[1] Full details are given in a paper on group coding pending for publication. It is not cited here, so as not to not violate the double blinded review process.

E. Design criteria: specifications

Specification 1. The verifier is offline during the interrogation (batch connectivity). Checking the integrity of a GoT when the verifier has permanent connectivity with the reader, and therefore with the tags, is straightforward. It is sufficient for individual tags to get authenticated by the verifier, who can then check simultaneous presence by using auxiliary data, *e.g.*, an identifier of the GoT. Therefore in this paper, we focus on offline solutions. In this case, the interactions of the verifier are restricted to: $i)$ broadcasting a challenge that is valid for a (short) time span and, $ii)$ checking responses from the tags of GoT (via intermediate readers) and compiling evidence of simultaneous presence.

Specification 2. A grouping-proof should be computed in a balanced, distributed way. No tag will assume the role of a "centralized" verifier. The tags of a group have similar hardware capabilities and the computation load per tag for generating a grouping-proof should be balanced.

Specification 3. Messages must include destination information (possibly private) to allow unicast/multicast communication. Although obvious, this is a common omission in many protocols. This is particularly important for anonymity: each message must contain information that allows tags to decide if they are the intended recipient.

F. Design criteria: assumptions

Assumption 1. The tags of a GoT are not compromised. Consequently tags of GoT can share private information. This does not mean that tags cannot be compromised; but if this happens then it is not possible to generate evidence that will support simultaneous presence in any meaningful way. Indeed if a tag T of G is controlled by the adversary, then T can prevent a grouping-proof from being generated by not participating actively, and conversely force a grouping-proof to be generated when it is not present via a proxy tag.

Assumption 2. Grouping-proofs apply to specific GoT: for a subgroup or extensions a different, independent proof should be sought. The tags of a GoT share the same private key: this is restricted to a specific GoT.

Assumption 3. Simultaneity is defined by valid interrogation intervals specified by the verifier. Grouping-proofs use session numbers, counters or timestamps provided by the verifier.

RFID communication is a sequential process and interrogation simultaneity can only be captured by an "exposure-time window": events are considered as happening simultaneously only if they take place within this window.

IV. An Anonymous Group-Scanning Protocol

Our grouping-proof is based on the design criteria in the previous sections and provides anonymity. The tags do not share any private information with the interrogating reader.

Requirements.

a) The verifier can check the integrity of a group of tags: that the tags were scanned simultaneously (during the same session defined by the activation time of the tags) within a time window defined by a counter T_s.
b) The reader can also check the integrity of the group, but does not share any private keys with the tags. The reader is not trusted and should not be able to access or even trace the tags beyond the lifetime of T_s.
c) During the lifetime of T_s, the reader can check if any tag is missing and obtain the identifiers of missing tags, but cannot generate a grouping-proof if tags are missing.
d) Tags can only generate random numbers and evaluate a one-way hash function: h.

The security objectives are twofold: (i) the verifier (owner) must be able to check the integrity of a group of shipped items if no items are missing, while (ii) the reader (carrier) must be able to recover the identifiers of missing tags if the shipment is compromised.

Protocol description

The verifier V stores for each group $G = \{tag_1, \ldots, tag_{n_g}\}$ of tags that it owns the tuple: $(T_s, K_g, \{K_i, ID_i\}_{i=1}^{n_g})$, where T_s is a counter value, K_g a group key, and K_i, ID_i the private key and identifier of tag_i (Section III-C). Each tag_i of G stores in non-volatile memory: ID_i, K_g, K_i, and a counter T_{s_i} that is initialized to the same value T_s for all tags of G. The reader R initially does not share any information with the tags of G.

The protocol is initiated by the verifier V who sends to the reader R a scanning request (T_s, T'_s, K_s), where T_s is a fresh value of a counter, $T'_s = h(K_s, T_s)$ is an authenticator and $K_s = h(K_g, T_s)$ is the session key. The protocol has two rounds—see Fig. 2.

Round 1. The reader R broadcasts to all tags in its range T_s, T'_s and sets a timer. Each tag_i in the range of R, computes $K_s = h(K_g, T_s)$, checks the integrity of T_s by checking $T'_s = h(K_s, T_s)$, and verifies that $T_s > T_{s_i}$. If any of these fail tag_i returns random values ("$*$" in Fig. 2). Otherwise, it updates the counter to T_s, draws a pseudo-random number r_i and computes its authenticator $r'_i = h(K_s, r_i)$. Then it sends r_i, r'_i to R and sets a timer. The received values r_i are used to identify (singulate) tags in this session. For every received r_i, the reader checks its integrity $r'_i = h(K_s, r_i)$. If this is correct, the value r_i is stored as part of the group G. Using these values, R computes a group session challenge $R_s = h(T_s, r_1, \ldots, r_{n_g})$ and its authenticator $R'_s = h(K_s, R_s)$. This round incorporates the randomness provided by the verifier's challenge T_s and the randomness provided by the tags r_i, which prevent replay attacks. The challenge T_s defines the scanning period for the verifier, and the simultaneity by defining the validity period of the nonces r_i.

Round 2. On timeout, the reader R broadcasts R_s, R'_s to all tags in its range. Each tag_i in the range of R that has not timed out, checks that $R'_s = h(K_s, R_s)$ and if so, computes:

$$M_i = h(K_s, r_i, ID_i),\ h(K_s, M_i) \oplus ID_i = M'_i \oplus ID_i,$$
$$P_i = h(K_i, r_i, R_s),\ P'_i = h(K_s, P_i),$$

sends $(M_i, M'_i \oplus ID_i, P_i, P'_i)$ to R and timeouts. The reader R computes $M'_i = h(K_s, M_i)$ and retrieves ID_i. Then, it

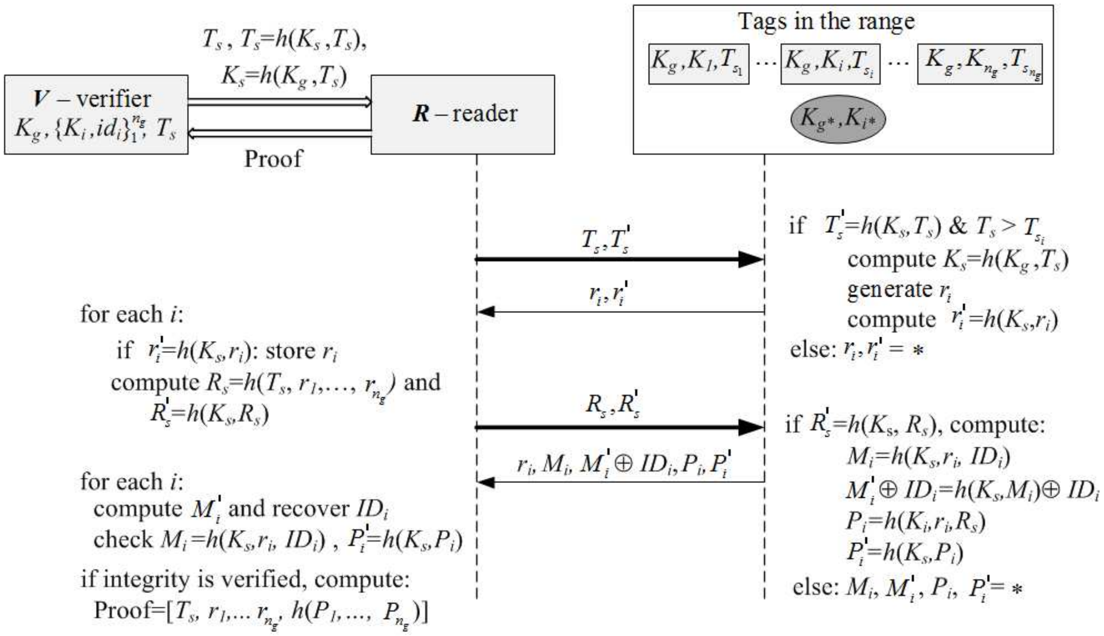
Tags in the range
Kg, K1, Ts1 … Kg, Ki, Tsi … Kg, Kng, Tsng
Kg*, Ki*
Ts , Ts'=h(Ks ,Ts),
Ks=h(Kg ,Ts)
V – verifier
Kg, {Ki, idi}1ng, Ts
R – reader
Proof
Ts,Ts'
if Ts'=h(Ks,Ts) & Ts > Tsi
compute Ks=h(Kg ,Ts)
generate ri
compute ri'=h(Ks,ri)
else: ri, ri' = *
ri, ri'
for each i:
if ri'=h(Ks,ri): store ri
compute Rs=h(Ts, r1,…, rng) and
Rs'=h(Ks,Rs)
Rs, Rs'
if Rs'=h(Ks, Rs), compute:
Mi=h(Ks,ri, IDi)
Mi'⊕ IDi=h(Ks,Mi)⊕ IDi
Pi=h(Ki,ri,Rs)
Pi'=h(Ks,Pi)
else: Mi, Mi', Pi, Pi'= *
ri, Mi, Mi'⊕ IDi, Pi, Pi'
for each i:
compute Mi' and recover IDi
check Mi=h(Ks,ri, IDi) , Pi'=h(Ks,Pi)
if integrity is verified, compute:
Proof=[Ts, r1,... rng, h(P1,..., Png)]

Fig. 2. Flows of the anonymous grouping-proof of integrity with missing tag identification; tags with group key K_g belong to the collection G while those with group key K_{g*} do not.

checks that $M_i = h(K_s, r_i, ID_i)$ and $P'_i = h(K_s, P_i)$. If these are correct, the reader verifies the integrity of the group by using the codewords ID_i. On timeout, if the list of identifiers is complete, it compiles the grouping proof: $(T_s, r_1, \dots, r_{n_g}, h(P_1, \dots, P_{n_g}))$ and sends this to the verifier. If the list is not complete then the reader R uses RS decoding to recover the missing tag identifiers and informs the verifier.

To validate the proof, the verifier V first uses T_s to get the values $(K_g, \{K_i, ID_i\}_{i=1}^{n_g})$ (one lookup). Then V computes $R_s = h(T_s, r_1, \dots, r_{n_g})$ using the values r_i given in the proof and the corresponding $P_i = h(K_i, r_i, R_s)$. Finally V checks that the value of $h(P_1, ldots, P_{n_g})$ in the proof is correct.

We shall assume that the keys K_g, K_i, K_s, the challenges T_s, R_s and the random numbers r_i, all have the same (bit) length κ, which is the *security parameter* of the protocol.

This protocol has just two rounds and only requires tags to be able to generate random numbers and compute a hash function.

V. Security analysis

A. Privacy

An adversary that physically tracks a group G of tags can determine which executions are linked to this group; this cannot be prevented. Similarly an adversarial reader that is authorized to inspect G can link the inspected tags. Unlinkability concerns the periods during which physical tracking or authorized inspection is interrupted.

Formally, unlinkability is defined in terms of an indistinguishability experiment $\mathtt{PrivK}^{link}_{\mathcal{A},\Pi}(\kappa)$, involving a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ and a grouping-proof Π. During the experiment $\mathcal{A}$ has access to simulated executions of Π, initialised with security parameter κ and random secret keys, and can interact with executions as specified in the threat model. Eventually $\mathcal{A}$ identifies two tags: tag^0 and tag^1 and is challenged with two grouping-proof interrogations $\mathtt{int}_0, \mathtt{int}_1$ involving tag^{b_0} and tag^{b_1}, respectively, b_0, b_1 random bits. $\mathcal{A}$ outputs bit b, with $b = 0$ if $\mathcal{A}$ decides that the tags are the same, and 1 otherwise. The output of the experiment is 1 if the adversary guessed right and 0 otherwise.

Definition. A grouping-proof provides *unlinkability* if: $\forall$ PPT $\mathcal{A}$: $\mathrm{Prob}[\mathtt{PrivK}^{link}_{\mathcal{A},\Pi}(\kappa) = 1] = \frac{1}{2} + \mathsf{neglible}$ (κ is the security parameter), where the probabilities are taken over the coin tosses of $\mathcal{A}$, the random bits and coins tosses used in the simulation of Π.

Proof of unlinkability(sketch). Every tag_i will update its counter T_{s_i} and draw a fresh pseudo-random number r_i after responding to the reader's challenge. Consequently the responses of tag^{b_0}, tag^{b_1} in interrogations $\mathtt{int}_0, \mathtt{int}_1$ are pseudorandom and cannot be distinguished with probability better than $1/2 + \mathsf{negligible}$.

B. Informal discussion of common attacks

Replay attacks. The use of the counter value T_s by the reader and the tags in the authenticators T'_s and r'_i prevents replay attacks: if an adversarial reader re-uses T_s, the tags that received this earlier will have updated their counter and will not respond. If a previous T_s was never sent to the tags, then the tags will respond (only this time) and a proof will be generated but this will not be accepted by the verifier (T_s is not valid). Similarly a replayed response r_i, r'_i for a previous counter value T_s will not be valid.

Impersonation attacks on tags are prevented by using private keys K_i. Impersonation attacks on a reader will not yield a valid proof: only readers that have access to the one-time challenge (T_s, K_s) of the verifier can interrogate the GoT. The $P_i\,(=h(K_i, r_i, R_s))$ from different sessions cannot be used to compose a proof since it involves the session nonces r_i of the interrogated tags and the challenge of the reader $R_s\,(=h(T_s, r_1, \ldots, r_{n_g}))$ that involves the time window specified by the counter T_s. Note that all tags set timers in Round 1 of the protocol, and will timeout if the challenge R_s is not received within the time window specified by the protocol.

De-synchronization attacks. If a protocol execution completes successfully then all tags will share the same counter value. No tag will accept a previously used T_s. Even if tags do not share the same counter value (*e.g.*, because of an interrupted interrogation), there are no synchronization concerns.

C. An anonymous RFID grouping code

As observed in Section II several RFID grouping codes that make it possible for an RFID reader to get the identifiers of groups of tags, including those of missing tags (forward error correction), have been proposed in the literature [15], [16], [17], [18], [19], [20], [21]. These codes *do not* address privacy issues (anonymity, unlinkability). The protocol in Section IV can easily be adapted to get an anonymous RFID grouping code by: (i) replacing the last flow of tag_i with: $(r_i, M_i, M_i' \oplus ID_i)$ (no P_i, P_i'); (ii) not having tag_i compute P_i, P_i'; and (iii) not having the reader check that $P_i' = h(K_s, P_i)$ and not computing the proof. Note that in the last flow of tag_i, the identifier ID_i is authenticated by M_i and encrypted by XORing it with the pseudo-random number M_i'.

VI. Conclusion

Several RFID grouping-proofs have been proposed in the literature. Many assume communication models, capabilities and design principles that either are not properly defined or are not practical. To the best of our knowledge, none of these make provision for missing tag identification. In this paper we address the group scanning problem in a strong adversarial setting. We define basic design criteria for anonymous group scanning and present an anonymous grouping-proof of integrity for groups G of RFID tagged objects that the verifier (a trusted entity): (i) can check it, (ii) can give an untrusted reader one-time access to the group G to recover missing tag information. The grouping proof has only two passes and provides strong anonymity. It resists replay, impersonation and de-synchronization attacks.

References

[1] EPC-Global, "Radio-Frequency Identity Protocols, Generation-2.V2. UHF RFID." Tech. Rep., April, 2015.

[2] G. Kapoor and S. Piramuthu, "Single RFID Tag Ownership Transfer Protocols," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 42, no. 2, pp. 164–173, 2012.

[3] F. G. Jorge Munilla and W. Susilo, "Cryptanalysis of an EPCC1G2 Standard Compliant Ownership Transfer Protocol," *Wireless Pers Commun*, no. 72, pp. 245–258, 2013.

[4] H. Liu, H. Ning, Y. Zhang, D. He, Q. Xiong, and L. T. Yang, "Grouping-proofs-based authentication protocol for distributed rfid systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 7, pp. 1321–1330, 2013.

[5] M. Burmester and J. Munilla, *Security and Trends in Wireless Identification and Sensing Platform Tags: Advancements in RFID*. IGI Global, 2013, ch. RFID Grouping-Proofs.

[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," *J. Netw. Comput. Appl.*, vol. 34, no. 3, pp. 833–845, May 2011. [Online]. Available: http://dx.doi.org/10.1016/j.jnca.2010.04.008

[7] A. Juels, "Yoking-proofs for RFID tags," in *PERCOMW '04: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 138–142.

[8] J. Saito and K. Sakurai, "Grouping proof for RFID tags," in *19th International Conference on Advanced Information Networking and Applications, AINA 2005.*, vol. 2, March 2005, pp. 621–624.

[9] A. Juels, "Generalized "yoking-proofs" for a group of RFID tags," in *MOBIQUITOUS 2006*, 2006.

[10] S. Piramuthu, "On existence proofs for multiple RFID tags," in *IEEE International Conference on Pervasive Services, Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing – SecPerU 2006*, IEEE. Lyon, France: IEEE Computer Society Press, June 2006.

[11] M. Burmester, B. de Medeiros, and R. Motta, "Provably Secure Grouping-Proofs for RFID Tags," in *CARDIS*, ser. Lecture Notes in Computer Science, G. Grimaud and F.-X. Standaert, Eds., vol. 5189. Springer, 2008, pp. 176–190.

[12] H.-H. Huang and C.-Y. Ku, "A rfid grouping proof protocol for medication safety of inpatient," *Journal of Medical Systems*, 2008.

[13] 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*, 2009.

[14] M. Burmester and J. Munilla, "Distributed group authentication for rfid supply management." *IACR Cryptology ePrint Archive*, vol. 2013, p. 779, 2013.

[15] Y. Sato, Y. Igarashi, J. Mitsugi, O. Nakamura, and J. Murai, "Identification of missing objects with group coding of RF tags," in *RFID, 2012 IEEE International Conference on*, April 2012, pp. 95–101.

[16] R. G. Gallager, "Low-density parity-check codes." *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[17] Y.-S. Su, J.-R. Lin, and O. K. Tonguz, "Grouping of RFID Tags via Strongly Selective Families," *IEEE Communications Letters*, vol. 17, no. 6, pp. 1120 – 1123, 2013.

[18] Y. Su and C. Wang, "Design and analysis of unequal missing protection for the grouping of rfid tags," *Communications, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2015.

[19] Y.-S. Su and O. K. Tonguz, "Using the Chinese Remainder Theorem for the Grouping of RFID Tags," *Communications, IEEE Transactions on*, vol. 61, no. 11, pp. 4741–4753, November 2013.

[20] Y.-S. Su, "Extended Grouping of RFID Tags Based on Resolvable Transversal Designs," *Signal Processing Letters, IEEE*, vol. 21, no. 4, pp. 488–492, April 2014.

[21] N. Ben Mabrouk and P. Couderc, "EraRFID: Reliable RFID systems using erasure coding," in *RFID, 2015 IEEE International Conference on*, April 2015, pp. 121–128.

[22] D. Paret, *RFID and Contactless Smart Card Applications*. John Wiley & Sons, 2005.

[23] International Organization for Standardization, "ISO/IEC 29192-1:Information Technology- Security Techniques - Lightweight cryptography - Part 1: General. ISO/IEC, 2012."

[24] V. Roca, M. Cunche, J. Lacan, A. Bouabdallah, and K. Matsuzono, "Simple Reed-Solomon Forward Error Correction (FEC) Scheme for FECFRAME," Tech. Rep., 2013.

[25] S. Bengio, G. Brassard, Y. Desmedt, C. Goutier, and J.-J. Quisquater, "Secure implementations of identification systems," *J. Cryptology*, vol. 4, no. 3, pp. 175–183, 1991.

[26] S. Mangard, T. Popp, and M. E. Oswald, *Power Analysis Attacks - Revealing the Secrets of Smart Cards*. Springer, 2007, vol. (ISBN: 0-387-30857-1).

[27] C. H. Kim, G. Avoine, F. Koeune, F.-X. Standaert, and O. Pereira, "The Swiss-Knife RFID Distance Bounding Protocol," in *ICISC*, ser. Lecture Notes in Computer Science, P. J. Lee and J. H. Cheon, Eds., vol. 5461. Springer, 2008, pp. 98–115.

[28] J. Munilla, A. Ortiz, and A. Peinado, "Distance Bounding Protocols with Void-Challenges for RFID," in *Workshop on RFID Security – RFIDSec'06*. Graz, Austria: Ecrypt, July 2006.

[29] D. N. Duc and K. Kim, "On the security of RFID group scanning protocols." *IEICE Transactions*, vol. 93-D, no. 3, pp. 528–530, 2010.