Predictive Protocol for the Scalable Identification of RFID Tags through Collaborative Readers

Rolando Trujillo-Rasua, Agusti Solanas, Pablo A. Pérez-Martínez and Josep Domingo-Ferrer

Department of Computer Engineering and Mathematics
Rovira i Virgili University. Catalonia, Spain.
E-mail: {rolando.trujillo, agusti.solanas, pabloalejandro.perez, josep.domingo}@ urv.cat

Abstract

Radio frequency identification (RFID) is a technology aimed at efficiently identifying products that has greatly influenced the manufacturing businesses in recent years. Although the RFID technology has been widely accepted by the manufacturing and retailing sectors, there are still many issues regarding its scalability, security and privacy.

With regard to privacy, the sharing of identification information amongst multiple parties is also an issue (especially after the massive outsourcing that is taking place in our global market). Securely and efficiently sharing identification information with multiple parties is a tough problem that must be considered so as to avert the undesired disclosure of confidential information.

In this article, we propose a private and scalable protocol for RFID collaborative readers to securely identify RFID tags. We define the general concepts of "next reader predictor" (NRP) and "previous reader predictor" (PRP) used by the readers to predict the trajectories of tags and collaborate efficiently. We also propose a specific Markov-based predictor implementation. By the very nature of our distributed protocol, the collaborative readers can naturally help in mitigating the problem of sharing identification information amongst multiple parties securely. The experimental results show that our proposal outperforms previous approaches.

Keywords: RFID, Scalable collaboration, Privacy, Secure information sharing, Trajectory analysis

1. Introduction

The amount of information that we have to store and share in order to produce, transport, and sell products is growing steadily. There is an urgent need for efficient technologies and protocols that allow the management of such a great amount of information in an efficient, secure and private way.

Radio frequency identification (RFID), a technology that was firstly used during the second world war to distinguish friendly airplanes from enemies, has become a must in the manufacturing and retailing sectors due to its ability to identify and track items rapidly and in parallel without the need for visual contact. This ability of the RFID technology has relegated bar codes to obsolescence. Moreover, RFID technology is especially suited for a variety of tasks, namely *assets tracking* (*e.g.* Air Canada decided to use this technology to control their food trolleys so as to reduce more than \$2 million in unexplained losses (17)), *manufacturing* (*e.g.* Boeing uses RFID to track parts as they arrive, and as they move from one shop to another within their facilities. Thus reducing errors and the need for people to look for parts (18)), *supply chain management* (*e.g.* Paramount farms, the largest producer of pistachio in the US, receives 50% of its production from a network of about 400 partners. The shipments are processed by using RFID that reduces processing times to up to 60% (21)), *retailing* (*e.g.* Walmart started to explore the RFID technology in 2003 and devoted at least three billion dollars to implement it (9)), and other applications such as *payments*, *security* and *access control*.

It is hard to say exactly how many RFID systems are already deployed worldwide. However, it is clear that these systems are becoming more popular with each passing day. Over 1.3 billion RFID tags were produced in 2005, and by 2010, that figure was expected to soar to 33 billion[1]. This rapid proliferation of RFID solutions strongly supports the paradigm of ubiquitous computing. In this scenario, billions of RFID tags will send information to thousands of RFID readers so as to enrich our interaction with the environment and make our processes more efficient and resilient. As a consequence of such a great proliferation, RFID readers must be able to identify, and distinguish, individual tags from sets of millions or billions of tags. Thus, the tags' identification protocol should be efficient, resilient and scalable. Further-

[1]According to a study of In-Stat (`http://www.in-stat.com`) - `http://www.instat.com/press.asp?Sku=IN0502115WT&ID=1545`

more, the identification procedure carried out by readers should guarantee the privacy of tags' holders as well as the security of the data stored in the tags' memories.

Take as an example the case of a company (Company A) devoted to the distribution of electronic products, namely desktop computers, laptops, displays, keyboards, etc. With the aim to improve the distribution chain, Company A attaches an RFID tag to each product, thus, it is possible to track them across multiple check points (*e.g.* factories, docks, resellers, and so on). The company requires the exchange of information between check points (*e.g.* RFID readers) to be **secure** and **reliable**, so as to avoid the loss of products. Also, the company wants the exchange of identification information between RFID tags and readers to be **private**, so as to prevent unauthorised parties from gauging the volume of products being shipped, exported or imported. Finally, the company wants this process of identification and tracking to be **efficient** and **scalable** (*i.e.* the identification of a product should be fast even in the presence of millions of possible products to be identified).

1.1. Contribution and plan of the article

The RFID technology can help to improve our processes by simplifying paperwork, speeding up the cataloguing of products and reducing costs. However, the massive use of this technology implies the management of billions of tags (which is a challenge) and might imply a thread in terms of security and, also, in terms of privacy.

In this article, we concentrate on these issues (*i.e.* Security, privacy and scalability in RFID systems)[2]. Specifically, our contributions are the following:

- We propose, describe and analyse a new protocol that allows the collaboration of multiple RFID readers so as to identify RFID tags securely and efficiently. The proposed protocol improves the efficiency and scalability of previous proposals, without sacrificing either security or privacy, in two ways:

[2]Note that tags implementing private protocols may coexist with tags that send their ID in clear text. In this article we assume that readers distinguish these two types of tags by means of physical procedures (e.g. using different frequencies for each type of tag). Thus, we concentrate on private tags only.

Figure 1: Basic components of an RFID system. From left to right: a back-end, RFID readers, and RFID tags. The back-end uses databases to store identification information. RFID readers are used to query RFID tags (that can take a variety of embodiments), retrieve their information, and forward it to the back-end through a wireless or wired channel. Note that in this simplest scheme RFID readers are used as relays and are not connected amongst them.

  - Predicting the next move of a tag. Thus, sending the identification information to the readers that really need it, and consequently reducing the information stored in the cache of the readers.
  - Predicting the estimated time of arrival (ETA) of tags, so that readers can search more efficiently in their caches.

- We show that the proposed protocol can naturally help mitigate the problem of exchanging identification information amongst multiple parties (which is especially useful for supply chains).

The rest of the article is organised as follows: In Section 2 we recall the basics of RFID systems and summarise the most relevant protocols devoted to the private identification of RFID tags. Next, in Section 3 we justify the need for new scalable identification methods and, we describe our proposal in detail. In Section 4, we compare a variety of identification methods and show that our proposal outperforms all previous ones. Finally, the article ends with a conclusion in Section 5.

2. Background

2.1. Basic scheme of RFID systems

Regardless of their operational frequency, materials or embodiments, RFID systems consist of three main components, namely tags, readers and back-ends (*cf.* to (1) for a brief survey on RFID and to Figure 1 for a graphical representation of the components and their basic relations/connections):

- **RFID tags** are small devices that can take a variety of possible shapes and embodiments (from stickers to small grains embedded in documents). The most basic RFID tags consist of a microchip and a metal coil. The microchip stores information, and is able to compute some simple operations; and the metal coil acts as an antenna that receives information from and sends information to readers. Optionally RFID tags can carry batteries, in this case they are called *active tags*. Otherwise, they are called *passive tags*. Passive tags are far more common than active tags because they are cheaper (*e.g.* a passive tag costs about $0.05, whilst an active tag might cost about $50 or more). Due to the fact that passive tags do not carry batteries, they harvest energy from the signal that they receive from readers and, consequently, they have very limited storage and computational power.

- **RFID readers** are devices utilized to retrieve information stored in RFID tags. In their simplest operation, readers emit a radio wave so that all tags in their cover range can power up and answer by broadcasting their embedded information (*i.e.* a set of bits[3]). After collecting the information, the readers forward it to a centralized computer (or *back-end*) along with their identification number and a timestamp.

- **Back-ends** are a set of databases connected to computers that receive, decrypt (if necessary), and manage the information collected by RFID readers about RFID tags. Back-ends store all the information required to identify RFID tags. Also, they can hold extra information about the products/items to which tags are attached.

2.2. Security, privacy and scalability in RFID systems

The main advantage of RFID systems is that tags can be read without the need for visual contact. However, this advantage might be also a problem due to the fact that unauthorised people with the right equipment might be able to interrogate tags and obtain their information without being detected. This kind of unauthorised access might lead to the disclosure of confidential information.

With the aim to solve this problem a wide variety of methods and protocols have been proposed. Designing private and secure, yet scalable, RFID identification protocols has been a big issue in recent years. The constrained computational resources on the tag's side make this task very challenging. In fact, protocols based on public key cryptography, that are widely accepted in electronic commerce, banking, or access control, are an unrealistic option

[3]In general these bits represent the electronic product code (EPC) of the item to which the RFID tag is attached.

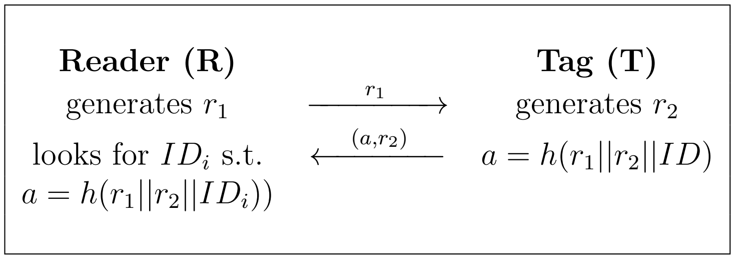

Figure 2: Improved randomized hash-locks protocol

for low-cost RFID tags due to their limited computational power. Therefore, most efforts have been focused on RFID identification protocols based on symmetric key cryptography. Such cryptographic systems are generally built upon some primitives like hash functions or block ciphers. Many efforts have been devoted to the development of light-weight cryptographic functions (14). Amongst them, the block cipher described in (4) is especially interesting since it may be implemented in very constrained RFID tags (*i.e.* it requires about 1600 logic gates). This means that current low-cost RFID tags might support symmetric-key cryptography.

Amongst all symmetric key identification protocols, the Improved Randomized Hash-locks (IRHL) protocol (10) is the most accepted one due to its strong privacy and security properties, and its low computational requirements on the tag's side (*i.e.* it only needs a pseudo-random number generator and a one-way hash function). In the IRHL protocol, for every tag's interrogation the reader generates a random number (*nonce*) r_1 and sends it to the tag. Upon reception, the tag generates another random number r_2 and computes the answer $a = h(r_1||r_2||ID)$ where ID is the secret identifier of the tag, $(\cdot||\cdot)$ is the concatenation operator, and $h(\cdot)$ is a one-way hash function. Finally, the reader receives from the tag the answer (a) and the nonce (r_2). With this information, the reader (or the back-end) determines the ID of the tag by performing an exhaustive search in its database looking for an identifier ID_i such that $a = h(r_1||r_2||ID_i)$. When that happens the tag is identified as ID_i. Figure 2 shows a graphical description of this protocol.

Although the IRHL scheme is a private and secure RFID authentication protocol, it cannot be used when the system contains a large number of tags (*e.g.* like in manufacturing processes) because for every tag identification there is a need for an exhaustive search in the database. As a result of this lack of scalability, several protocols have been designed to reduce the linear complexity in the identification process of the IRHL protocol.

Tree-based protocols, such as the one proposed by Molnar and Wagner (MW) (13), may be considered an alternative to IRHL in terms of scalability. The MW protocol achieves a time complexity in the identification process of $O(d \times \log_d^N)$ where N is the number of tags in the system and d is the branching factor of the tree that is used to store the tag's identifiers. When tree-based protocols are used, each tag stores a set of keys that uniquely identify it and, also, it must share, at least, one key with every other tag in the system. This sharing of private identification information might lead to undesired privacy leaks if a given number of tags are compromised[4]. The larger the number of compromised tags, the greater the risks for privacy (3).

Similarly to tree-based protocols, group-based protocols, such as the one described in (2), try to reduce the computational cost related to the secure identification of tags. In this case, tags are randomly assigned to groups. Consequently, each tag stores its own ID and the ID of the group to which it belongs. During the identification process, tags first send the group ID and then their own ID. In this way, the identification of tags is simplified because each tag has to be identified within its group instead of amongst all possible tags. Although group-based protocols could perform even better than tree-based protocols (2), if a tag is compromised, the whole group to which the tag belongs is compromised too. Thus, from the scalability perspective group-based protocols are very efficient, but with regard to privacy, they are not a good choice.

In the above protocols, the use of a single reader is assumed[5]. However, in 2007, Solanas *et al.* (15) introduced the idea of using multiple collaborative readers to make the identification process scalable whilst maintaining the high level of privacy of the IRHL scheme. Their proposal is aimed at efficiently identifying tags in applications where each tag must be continuously monitored whilst it remains in the system. This implies that readers must cover the whole system. Under this assumption, tags are constrained to move along neighbour readers[6] and therefore, neighbour readers collaborate in order to guarantee efficiency during the identification process. Efficiency

[4]Even a single compromised tag might lead to a privacy leak

[5]This reader being connected to a back-end that is responsible for the computation of the identification operations. Note that, generally, readers are considered as simple relays that forward identification information to back-ends.

[6]Two readers are said to be neighbours if their cover areas are not disjoint.

is achieved by means of the so-called *reader's cache*, which is defined as a storage device where a reader saves identification data of tags[7]. The protocol reduces the size of the readers' cache by considering that only the closest reader to some tag and its neighbours must store the identification information of this tag. By reducing the size of the cache, the identification procedure becomes more efficient. Despite the benefits in terms of computational cost provided by this protocol, assuming that readers are able to compute their accurate distance to tags is a bit unrealistic.

With the aim to solve the practical problems of (15), a novel protocol was proposed in (19) aiming at identifying RFID tags efficiently without requiring special skills to readers (*e.g.* readers-tags distance computation). This protocol is based on the Solanas *et al.* protocol (15) but it has significant differences. A parameter $p \in [0, 1]$ is used to reduce the size of the readers' cache. The lower p the smaller the cache. Theoretically, when $p = 1$ the size of the readers' cache roughly equals the Solanas *et al.* protocol, whilst when $p = 0$ the minimum size is reached (at the cost of increasing the number of messages sent between neighbour readers). The idea is that p may be tuned up according to the application needs so as to favour a reduction of computations or bandwidth usage. Also, the protocol does not impose any constraint on the cover areas of the readers, or on their communications architecture.

In the context of using multiple readers (connected to a centralised back-end), Fouladgar and Afifi (5) point out that, in many applications, tags are usually queried by the same set of readers. Therefore, they propose to cluster tags according to the readers that identify them most often. This idea improves on the group-based proposals in the sense that tags are not randomly assigned to groups, but intelligently clustered according to the spacial location of the readers that identify them. By doing so, when a reader receives a tag's response, it first performs a search on the group of tags that it usually identifies. If it does not succeed, an exhaustive search is performed over the whole set of tags' identifiers. The problem of this proposal is that tags may have a long life-cycle and move through a wide variety of readers. In this scenario, the protocol could scale as bad as previous protocols based on symmetric key cryptography (10).

[7]This cache can be either an external database securely connected to the reader or a database internally managed by the reader itself.

Figure 3: Intuitive location of several identification protocols in the privacy-scalability plane. The top-right quadrant is where private and scalable protocols lie.

Recently, a new protocol based on collaborative readers was proposed by Trujillo and Solanas (20). Similarly to previous protocols (15, 19), tags' data are stored on the cache of the readers that contain these tags in their cover area. However, this proposal is able to perform at least 50% better than the Solanas et. al. (15) and the Fouladgar and Afifi (5) proposals in terms of computational cost. The improvement is due to the use of a heuristic that allows readers to estimate the most probable previous location of tags. By doing so, once a reader could not identify a tag using its own cache, it asks for the information to another reader that is able to identify it with a given confidence. In addition to the reduction of the computational cost, the proposal has significant improvements in terms of flexibility and usability with respect to other proposals like (19). It is remarkable that readers do not need either to cover the whole system or they have to rely on a neighbourhood relationship.

Some protocols are able to achieve good scalability and privacy by periodically "refreshing" the identification information stored in tags. These protocols add to the identification phase an "update" phase. Thus, the protocols assume that tags are able to change the information they store and recompute all necessary parameters of the protocol. Currently, this assumption is unrealistic for low-cost passive tags. Hence, we focus on the previously explained protocols and propose a new predictive protocol that improves the scalability of the aforementioned protocols whilst guaranteeing the same level of privacy of the IRHL scheme. Figure 3 depicts an intuitive distribution of the aforementioned protocols in the privacy-scalability plane[8].

3. Our proposal

3.1. Rationale

The IRHL protocol (10) is a private and secure solution for the identification of RFID tags. However, the computational cost of identifying a tag is

[8]This Figure is not intended for an exact evaluation but for an intuitive/approximative yet illustrative description of the relation between privacy and scalability of the aforementioned protocols.

linear $O(N)$ with the number of tags (N). Thus, it can hardly be used when the number of tags in the system is large.

Using collaborative readers to improve the scalability of the IRHL protocol has proven to be effective (*cf.* (15), (19) and (20)). The main reason to use several collaborative readers is to distribute the identification information amongst all the readers in the system. In this way, the exhaustive search that would be required in a centralised scheme (with a single cache/memory) is applied to the individual cache/memory of each individual reader (15), and the computational cost is distributed amongst the k collaborative readers leading to a cost $O(\alpha\frac{N}{k})$, where α is the degree of redundancy[9] in the caches of the readers.

Thanks to the Trujillo and Solanas protocol described in (19), the redundancy of the caches can be significantly reduced at the cost of increasing the number of messages sent between the collaborative readers. And thanks to the use of the information of the trajectories of the tags (20), the amount of sent messages can also be reduced. However, due to the fact that $N \gg k$ the computational cost is still linear in all cases.

We leverage the idea from (20) of using the trajectory information of the tags to determine the readers that might help identify a tag when a given reader is not able to do it itself. In addition, we also predict the next position of a tag and the time at which the tag will reach a given reader. Thus, readers can share spatio-temporal information about tags, so as to identify them without the need for an exhaustive search in their caches, as we show next. To the best of our knowledge this is the first time that a predictive algorithm for collaborative readers based on trajectories is used to successfully speed up the identification process of RFID tags.

3.2. Cache structure

In previous proposals (15), (19) and (20), the readers' cache contains tags' identification data but lacks information about the expected time at which the tags might next be identified by a reader or where they were identified in the past. Assuming that it is possible to approximately know the instant at which a tag will be identified by a given reader, it is greatly beneficial to use this spatio-temporal information to speed up the searching process in the readers' cache. Therefore, we propose to structure the readers' cache

[9]The degree of redundancy highly depends on the neighbours topology (15).

Cache of Reader 512				
ETA	Tag ID	Previous Reader	Next Reader	First Time
2011-07-28 11:31:38	90876534	1012	201	Yes
2011-07-28 11:41:33	10311299	1011	1201	No
...	...	...	...	...
2011-07-30 22:01:08	21134211	1012	201	No

Table 1: Example of the cache of a reader.

as an ordered list where the expected time of arrival (ETA) is the ordering criterion.

Definition 1 (Cache). *Given the set of tags $\mathcal{T}$ and readers $\mathcal{R}$ in the system, the cache of a reader $R \in \mathcal{R}$ consists of a sequence of ordered tuples*

$$C(R) = < t_1, ID_1, R^{ID_1}_{prev}, R^{ID_1}_{next}, Y|N >, \cdots ,$$

$$, \cdots , < t_N, ID_N, R^{ID_N}_{prev}, R^{ID_N}_{next}, Y|N >$$

where the order is given by the timestamps $t_1 \leq \cdots \leq t_N$. The tag identifiers $ID_i \in \mathcal{T}$, $\forall\ 1 \leq i \leq N$, and $R^{ID_i}_{prev} \in \mathcal{R}$ and $R^{ID_i}_{next} \in \mathcal{R}$ are the reader that sent the ID_i to R and the reader that will receive the ID_i from R respectively. $Y|N$ is used as a flag to show whether the tag has been already identified by this reader.

From the above definition it can be observed that our protocol will use the spatial information about the trajectory of the tags (*i.e.* to predict which reader will be the next reader to receive a given tag), and the temporal information (*i.e.* to predict when a given tag will be read in the future by the next reader). Table 1 is an example of the cache of a reader. In this example, the reader R_{512} expects to receive the tag $T_{90876534}$ from reader R_{1012} at time *2011-07-28 11:31:38*, and will forward the identification information to the next reader R_{201}. Also, it can be seen that the tag has not been identified by the reader yet.

By using this ordered cache, when a tag response arrives at a given timestamp t, a reader is able to optimize the searching process in its cache by first considering the tags that it expects to identify at a timestamp t' close to t. Note that if the ETA is accurate, the identification of tags might be very fast. The better the prediction the faster the identification process. In the worst case the computational cost is linear $O(n)$, where n is the number of identifiers in the cache of the reader.

3.3. Trajectory prediction algorithms

The idea of using heuristics to speed up the identification process has been recently introduced by Trujillo and Solanas in (20). Although the results achieved by (20) are very good, the heuristics that are used consider the movement of all tags globally (*i.e.* they look for global trends instead of predicting the moves of a single tag by using a trajectory predictor algorithm). In (20) the heuristics are not used to inform readers about the most probable position of tags before the identification. They are only used to find the reader that might have the information about a tag once the reader that should identify it fails to do so.

We propose to use trajectory predictors in a broader manner, so as to be able to inform readers about which tags they will receive and when before they actually receive them.

In general, a trajectory is understood as a timely ordered set of consecutive points ($\mathcal{P}$) defined in an n-dimensional space ($\mathcal{S}$). However, due to the fact that we can only control the location of the tags when they are detected by a reader, we define our concept of trajectory as follows:

Definition 2 (Trajectory)**.** *Given a set of readers $\mathcal{R}$ and tags $\mathcal{T}$. The trajectory of a tag $T_i \in \mathcal{T}$ is defined as a sequence*

$$S_i = < t_1, R_1 >, < t_2, R_2 >, \cdots, < t_{s(i)}, R_{s(i)} >$$

where $s(i)$ is the size of the sequence, $t_1 < t_2 < \cdots < t_{s(i)}$ are timestamps and, $R_j \in \mathcal{R}\ \forall 1 \leq j \leq s(i)$ are the readers that identified the tag T_i at the timestamp t_j.

When a tag arrives at the cover area of a reader, the reader tries to identify it by applying the already explained IRHL protocol. During the identification process two situations could arise:

Figure 4: Illustration of the identification success of the tag T_i by reader A. After successfully identifying T_i, the reader A applies a Next Reader Predictor (NRP) to predict the next reader (in this case, reader A decides that reader B is the best candidate) and sends the T_i identification information to reader B. The reader B stores the information about T_i in its cache so as to be able to identify it (if necessary).

1. **Identification success**: The reader finds the identification information of the tag in its cache and can identify it. Then it has to decide to which reader should this information be forwarded (See the example of Figure 4).
2. **Identification failure**: The cache of the reader does not contain the identification information of the tag and the reader cannot identify it. The reader has to decide to which other reader ask for help (See the example of Figure 5).

In the first case (*identification success*), after properly identifying a tag, the reader will proceed by using an algorithm (*i.e.* the "Next Reader Predictor" (NRP)) to determine which reader will be the next one to which the tag will move. Once this next reader is determined, the current reader sends the identification information of the tag to that reader. An NRP can be defined as follows:

Definition 3 (Next Reader Predictor (NRP))**.** *Let T_i be a tag of the system and let $S_i = < t_1, R_1 >, < t_2, R_2 >, \cdots, < t_j, R_j >$ be its trajectory of size (j). An* NRP *is a polynomial-time algorithm (let us call it $\mathcal{A}_{next}$) that on input T_i and S_i outputs the pair $< t_{j+1}, R_{j+1} >$.*

$$\mathcal{A}_{next}(T_i, S_i) \longrightarrow < t_{j+1}, R_{j+1} >$$

This output pair means that it is expected that the tag T_i will be identified at time $t_{j+1} > t_j$ by the reader R_{j+1}.

Note that the result of the NRP is correct only with a probability that highly depends on the utilised algorithm and the degree of regularity of the movement of tags [10]. Thus, if the prediction is wrong, the reader which is

[10] It is apparent that in a chaotic system where no regularities exist, the prediction of the next move of a tag would be extremely inefficient.

Figure 5: Illustration of the identification failure. Reader C tries to identify T_i but it fails because it has not got the information in its cache. It uses a PRP to decide which reader to ask for help (in this case C asks A).

currently identifying the tag T_i will forward the identification information to a wrong reader. As a consequence, when that tag reaches the next reader, the latter will not be able to identify the tag (because the identification information will not be in its cache) and will need the help of other readers to do so (this is the second case enumerated above).

In the second case (*Identification failure*), when a reader cannot identify a tag, it proceeds by using an algorithm (*i.e.* the "Previous Reader Predictor" (PRP)) to identify the reader that might have identified the tag previously and might have the identification information of the tag. A PRP can be defined as follows:

Definition 4 (Previous Reader Predictor (PRP))**.** *Let $\mathcal{T} = \{T_1, \cdots, T_N\}$ be the set of tags in the system and let $\mathcal{S} = \{S_1, \cdots, S_N\}$ be the set of trajectories of the tags in $\mathcal{T}$ until a given time t. Let $\mathcal{T}^{\mathcal{R}} \subset \mathcal{T}$ be the subset of tags known by reader $\mathcal{R}$ and let $\mathcal{S}^{\mathcal{R}} \subset \mathcal{S}$ be the trajectories of the subset of tags known by reader $\mathcal{R}$. A* PRP *is a polynomial-time algorithm (let us call it $\mathcal{A}_{prev}$) that on input a reader $\mathcal{R}$ and, a set of trajectories of tags $\mathcal{S}^{\mathcal{R}}$, outputs the sequence of k readers $R_1, R_2, \cdots, R_k$ that are candidates to be the previous reader that identified a tag.*

$$\mathcal{A}_{prev}(\mathcal{R}, \mathcal{S}^{\mathcal{R}}) \longrightarrow < R_1, R_2, \cdots, R_k >$$

Remark 1. *The order of the sequence of candidate readers depends on the specific implementation of the predictor. However, the following condition must hold:*

$$P(success|R_1) \geq P(success|R_2) \geq \cdots \geq P(success|R_k)$$

This means that R_1 has more chances of being the actual previous reader than R_2, etc.

Figure 6: Conceptual logical flow of the protocol.

Note that in this section we have defined the theoretical concept of NRP and PRP algorithms. However, the specific implementation of these algorithms would highly affect the performance of the whole system. In the next sections we will give details on the implementations that we have used for our experimental analysis.

3.4. Our protocol

We define our protocol as a distributed algorithm in the context of a set of collaborative readers $\mathcal{R}$ that share identification information on a number of tags $\mathcal{T}$. For the sake of completeness, in addition to $\mathcal{R}$, we consider a special reader $\mathcal{O}_R$ that acts as an oracle (*i.e.* it has the same role of classical back-ends that have the information of all tags in the system). The oracle $\mathcal{O}_R$ is able to identify any tag in $\mathcal{T}$. In our collaborative context, it should be understood as the *"last resort"* to identify a tag if all the other mechanisms fail. Thanks to the use of an oracle no false negative identifications take place. Notwithstanding, the computational cost associated to the identification of tags by the oracle is very high. Thus, it must be used only when all the other identification procedures fail.[11]

Algorithm 1 shows a pseudocode description of our protocol and Figure 6 depicts the logical flow of the proposal. Our protocol works as follows: The reader R, that receives an identification message from an unidentified tag T at time t, tries to identify it by following the IRHL scheme described in Section 2 but using the identification information stored in its own cache only *(lines 1 to 10 in Algorithm 1)*). In order to perform this identification efficiently, the reader uses the cache structure described above. First, it tries to identify T as one of the tags that were expected to arrive at time t. If the tag is not identified amongst these candidate tags, the reader tries with tags that were expected to arrive a bit later at time $t + 1$ and a bit earlier at time $t - 1$, and so on. Searching in this way, if the ETA of T was properly predicted and forwarded, T is identified almost instantly. However, if the prediction was wrong, the reader R might need to search over all its cache.

[11]Note that this situation might happen rarely and probably it would be caused by a communication failure amongst the collaborative readers or by an active attack. In normal conditions, the oracle should not be used.

Algorithm 1 Main protocol

```
Require: A tag $T$ to be identified by a reader $R$ at time $t$;
Require: The set of tags' trajectories $\mathcal{S}^{\mathcal{R}}$ known by $R$;
Require: The cache of reader $R$, $C(R)$.

    - - Try to identify T using the local cache
 1: for all $t' \in \{t, t+1, t-1, t+2, t-2, \cdots\}$ do
 2:   for all $T_i \in C(R)$ with $ETA = t'$ do
 3:     if $T$ is identified as $T_i$ then
 4:       if $T_i$ is a "first time" tag then
 5:         Call New_Tag ($R$, $T_i$, $t$, $R^{T_i}_{prev}$, $S_i$)
 6:       end if
 7:       return (Tag identified correctly);
 8:     end if
 9:   end for
10: end for
    - - Try to identify the tag with other readers' help
11: for all $R' \in < R_1, R_2, \cdots, R_k > \longleftarrow \mathcal{A}_{prev}(R, \mathcal{S}^{\mathcal{R}})$ do
12:   Call Help_Identify ($T$,$R'$)
13:   if $R'$ identifies $T$ then
14:     $R$ receives $< t', T_i, R^{T_i}_{prev}, R^{T_i}_{next} >$ from $R'$.
15:     Call New_Tag ($R$, $T_i$, $t$, $R'$ and, $S_i$).
16:     return (Tag identified correctly);
17:   end if
18: end for
    - - (last resort) Ask the Oracle
19: if $\mathcal{O}_R$ identifies $T$ then
20:   Call NewTag ($R$, $T$, $t$, $\mathcal{O}_R$, $\emptyset$).
21:   return (Tag identified correctly);
22: end if
23: return Invalid tag $T$ found
```

If T is identified, R checks whether it is the first time that this tag enters its interrogation zone (*i.e.* it is a "first time" tag) and, if it is, it calls the procedure *New_Tag*$(R, T_i, t, R^{T_i}_{prev}, S_i)$ and the identification finishes. If the tag is not a "first time" tag, the identification procedure simply finishes.

If the identification information of T was not properly forwarded to R[12], it will search over all its cache and will not be able to identify T. In this situation, it has to ask for help to the other collaborative readers that might have the information it needs (*lines 11 to 18 in Algorithm 1*). To do so, R calls the PRP algorithm $\mathcal{A}_{prev}(R, S)$ so as to obtain a list of readers that may have information about T. For each reader R' in the list returned by $\mathcal{A}_{prev}$, the procedure *Help_Identify* (T,R') is called. If this procedure succeeds in identifying T, the collaborative reader that succeeds sends the tuple of its cache that contains the information about T (*i.e.* $< t', T_i, R^{Ti}_{prev}, R^{Ti}_{next} >$) to R. By using the information in this tuple the identification process correctly finishes after calling the procedure *New_Tag*(R, T_i, t, R', S_i).

Finally, if no reader R' can identify T, R asks the oracle $\mathcal{O}_R$ (*lines 19 - 23 in Algorithm 1*). If $\mathcal{O}_R$ cannot identify T, the latter can be considered an illegitimate tag[13]. Otherwise, R finishes successfully the identification process by calling procedure *New_Tag* (R, T, t, $\mathcal{O}_R$, $\emptyset$).

Algorithm 2 New_Tag

```
Require: A reader $R$ that has identified a tag $T_i$ at time $t$;
Require: The reader $R^i_{prev}$;
Require: The trajectory $S_i$ of $T_i$.

1: $R$ asks $R^{T_i}_{prev}$ to remove $T_i$ from its cache;
2: $R$ predicts the next reader and ETA $< t_i, R^{T_i}_{next} >= \mathcal{A}_{next}(T_i, S_i)$;
3: $R$ asks $R^{T_i}_{next}$ to insert the record $< t_i, T_i, R, null, Y >$ in its cache;
4: $R$ removes the record about $T_i$ from $C(R)$ (if it exists);
5: $R$ inserts the record $< t, T_i, R^{T_i}_{prev}, R^{T_i}_{next}, N >$ into $C(R)$;
6: $R$ adds $< t, R >$ to the $T_i$'s trajectory ($S_i$);
```

The main protocol described in Algorithm 1 uses two procedures (*i.e.* *New_Tag* and *Help_Identify*) to update the state of the caches of other collaborative readers and to identify tags from which the identifying reader has no information.

[12]Note that this might happen due to a wrong prediction of the next reader by the previous reader.

[13]In this case, the proper actions are to be taken, namely raise an alarm, locate and eliminate the tag, etc.

The *New_Tag* procedure, described in Algorithm 2, is called when a reader R determines that a newly identified tag, T_i, has entered its interrogation zone for the first time (*i.e.* it is a "First time" tag) and thus, T_i's trajectory must be updated. In this case, R sends a message to the previous reader $R^{T_i}_{prev}$ that identified T_i so as to let it remove the information it has about T_i[14] (note that, when $R^{T_i}_{pre} = \mathcal{O}_R$ this message is not sent). Then, R uses an NRP to determine the next reader that will be visited by T_i and sends a messages to it to let it insert the tuple $< t_i, T_i, R, null, Y >$ in its cache (this way, when the tag reaches this reader, it will be able to identify it efficiently). Finally, the record corresponding to T_i in $C(R)$ is updated with proper information about the next reader $< t, T_i, R^{T_i}_{prev}, R^{T_i}_{next}, N >$.

Algorithm 3 Help_Identify

```
Require: $T$ a tag to be identified by a reader $R$;
 1: Determine $t_{old}$ the oldest timestamp in $C(R)$;
 2: for all $t' \in \{t_{old}, t_{old}+1, t_{old}+2 \cdots\}$ do
 3:     for all $T_i \in C(R)$ with $ETA = t'$ do
 4:         if $T$ is dentified as $T_i$ then
 5:             $R$ asks $R^{i}_{prev}$ to remove $T_i$ information from its cache;
 6:             return $< t', T_i, R^{T_i}_{prev}, R^{T_i}_{next} >$
 7:         end if
 8:     end for
 9: end for
10: return $< null >$ (Tag not identified)
```

The *Help_Identify* procedure, described in Algorithm 3, is called when a reader R cannot identify a tag with the information stored in its cache. This procedure is executed by the readers that collaborate with R. Due to the fact that these collaborative readers might have seen the unknown tag quite in the past, they start searching tuples in their caches whose timestamps are old. If a collaborative reader R' identifies T as T_i it sends a message to $R^{T_i}_{prev}$ in order to let it remove the information on T_i from its cache. Finally, R' returns the tuple about T_i stored in its cache.

[14]This information is no longer necessary and removing it from the cache speeds up the identification procedure

3.5. Practical implementation of the predictors

Previously, in Section 3.3, we have theoretically defined the concepts of Next Reader Predictor (NRP) and Previous Reader Predictor (PRP). In this section, we provide the reader with a practical implementation for each of these predictors.

3.5.1. Next reader predictor

We propose the use of a location prediction algorithm based on a Markov model (16). A Markov-based predictor of order k, $(O(k))$, is defined over the sequence of the last k locations of a given moving entity. Let $L = \ell_1, \cdots, \ell_n$ be the location history of a given entity and let $L(i,j) = \ell_i, \cdots, \ell_j$ be a subsequence of L. Let X_i be the random variable that represents a location at time t. Then, the Markov assumption is that:

$$\Pr(X_{n+1} = x | X_1 = \ell_1, \cdots, X_n = \ell_n) =$$
$$\Pr(X_{n+1} = x | X_{n-k+1} = \ell_{n-k+1}, \cdots, X_n = \ell_n) \tag{1}$$

And that for every $i \in \{1, 2, \cdots, n-k\}$:

$$\Pr(X_{n+1} = x | X_{n-k+1} = \ell_{n-k+1}, \cdots, X_n = \ell_n) =$$
$$\Pr(X_{i+k} = x | X_{i-k} = \ell_{n-k+1}, \cdots, X_{i+k-1} = \ell_n) \tag{2}$$

Simply stated, Equation 1 says that the probability of being in a given location depends on the previous k locations only, whilst Equation 2 says that this probability is time independent. Therefore, this probability can be represented by a transition matrix M labelled with all possible sequences of locations of size k:

$$\Pr(X_{n+1} = x | X_1 = \ell_1, \cdots, X_n = \ell_n) =$$
$$M(L(n-k+1, n), L(n-k+1, n) || x) \tag{3}$$

And the value of $M(a,b)$ may be estimated by

$$M(a,b) = \frac{N(a,L)}{N(b,L)} \tag{4}$$

Where $N(s_1,\ s_2)$ is the number of times the subsequence s_1 occurs in the sequence s_2.

In our protocol, locations are represented by the readers $\mathcal{R}$ and a next reader predictor (NRP) is only used by readers $\mathcal{R}$ once they realise that a tag T is in their interrogation zone. Thus, the last location of T is the current reader R, *i.e.* $\ell_n = R$. Therefore, we believe that a reader could be able to implement a Markov-based predictor of order 1 or 2 using a reasonably small amount of memory. In addition, counting the number of times that a tag is identified by a reader after having been identified by another reader can be easily done when calling the *New_Tag* procedure described in Algorithm 2. Our Markov-based predictor is computationally efficient. It has a logarithmic computational cost with respect to the number of readers $\mathcal{R}$.

Regarding the time prediction, we use a very simple approach. Let t_m be the average time in which a tag T is identified by two consecutive readers. Let t be the current time in which T is identified by a reader. We estimate that the next reader will identify T at time $t + t_m$. Note that the readers store, share and update t_m. To update the value of t_m, the reader applies the following equation:

$$t_m = \frac{t_m \times (c-1) + t - t_{last}}{c},$$

where c is the number of times that the tag has been identified and t_{last} is the last time in which that tag was identified.

3.5.2. Previous reader predictor

The idea of using a previous reader predictor was recently proposed by Trujillo and Solanas in (20). We propose to use the same heuristic proposed in (20) because it has been shown to be efficient and provides good results.

In a nutshell, the proposed predictor works as follows: When a reader R_i identifies a tag, it increments a counter $G(R_i, R_j)$, where R_j is the last reader that identified that tag. By doing so, when $\mathcal{A}_{prev}(\mathcal{R}_i, \mathcal{S}^{\mathcal{R}_i})$ is called, it outputs the sequence,

$$R_1, R_2, \cdots, R_k$$

such that

$$G(R_i, R_1) \geq G(R_i, R_2) \geq \cdots \geq G(R_i, R_k)$$

The computational cost of $\mathcal{A}_{prev}$ is logarithmic with respect to the number of readers $\mathcal{R}$. Note that there is no need for sorting the output list every time the algorithm is called (*i.e.* this might lead to a computational complexity $O(|\mathcal{A}| \log |\mathcal{A}|)$). On the contrary, the list could be stored already sorted and

Figure 7: Illustration of our example.

simply updated after increasing the value of $G(R_i, R_j)$ for any pair of readers R_i and R_j.

Note that the PRP is essentially a “global” predictor in the sense that it is based on the information of the trajectories of multiple tags. Consequently, it can be seen as a trend analyser (*e.g.* if most of the tags that are identified by a reader R_y move to a reader R_x, when the reader R_x uses the PRP, the first result will be R_y). On the contrary, the NRP described in the previous section is essentially “local” in the sense that it only depends on the information of a single tag.

3.6. Sharing private information with our proposal

Securely sharing information amongst multiple parties is a problem that has been already studied in other areas such as secure multi-party computation, private information retrieval, etc.

The main goal of this article is to propose a new protocol that improves the scalability of previous RFID identification proposals by means of collaborative readers. However, our protocol can be used to lessen the problem of sharing identification information amongst multiple parties, also.

The collaboration amongst multiple readers is a fundamental feature of our proposal. By means of this collaboration, readers can exchange identification information and distribute the whole identification database amongst them. By doing so, each reader has information about the tags/products that are in its interrogation zone and the ones that are going to be there in the near future. By the very nature of our proposal, readers only have the information they must have (*i.e.* the one they need to identify the tags in their surroundings). Let us illustrate this property of our proposal with an example.

Example 1. *Let us suppose that we have two manufacturing companies (e.g. Company A and Company B). Company A is selling products to Company B. Both companies use RFID technology so Company A has a reader (Reader A) that monitors the products that are sent out of the manufacturing plant and, Company B has a reader (Reader B) that monitors the products that enter the manufacturing plant.*

For security reasons, Company A uses a private identification protocol like the improved randomized hash-locks. Thus, in order to identify the products that Company A sells to Company B, the latter must have access to their identification information.

In the scenario described in the previous example, Company B should have access to the identification information of the products that it bought to Company A. In the case of the classical IRHL scheme, Company A should grant access to its back-end to Company B, so as to let it identify the products. However, if Company B has access to the back-end of Company A, the former can obtain extra information about the latter (note that this information could be considered confidential because it could lead to the disclosure of private information such as the state of the stock).

By using our protocol, Company B will only have access to the information sent by Reader A to Reader B. This information is enough to identify all the products that have been sent to Company B and it cannot obtain any extra information from it.

In our example (illustrated in Figure 7) the back-end of Company A has information about all the stock (26 products in this case). Reader A has information about the products that are passing through it (*i.e.* products 1 to 9) and are being sent to Company B. The information of Reader A is timely and properly forwarded to Reader B (in our example, Reader B has information about the tags that it has already detected (1,2, and 3) and the ones that it is going to receive (4 - 9). The information received by Reader B is also forwarded to the back-end of Company B.

Clearly, this property is not the most important of our proposal. However, it can help to reduce the overheads related to the management of identification information transparently.

4. Experimental results and evaluation

With the aim to assess the correctness and efficiency of our protocol with respect to other state-of-the-art proposals, we have obtained experimental results based on the simulation of those proposals by using data sets of trajectories.

A data set of trajectories contains a historical log with all the identification events produced by readers during the identification of tagged objects in a given scenario. Thus, with these data sets, it is possible to determine the precise moment in which a tagged object was identified by a given reader in an exact location. By using these data sets of trajectories, whether real or synthetic, we are able to measure the performance of our proposal, in terms of computational cost and bandwidth usage, and compare it to others without the need for an expensive and very time consuming implementation of real prototypes.

Notwithstanding, obtaining real data sets of trajectories of RFID tagged objects moving through, for example, supply chains is very difficult (*i.e.* these data are generally kept by private companies that are quite reluctant to share them). Hence, the use of synthetic data obtained by means of simulation is a common practice (6) (7) (8). However, a synthetic data set might fall short of capturing the real complexity of the motion of objects. With the aim to lessen this problem and in order to perform a comprehensive comparison of our proposal with previous ones, we use two different data sets of trajectories:

1. A synthetic data set generated by simulating the movement of tagged objects in supply chains. This data set has been generated by using techniques proposed in previous articles (6) (7) (8), which deal with moving objects in supply chains.
2. A real data set consisting of a historical log of the movement of wireless cards through several access points at Dartmouth College (11). This real data set of trajectories captures the movement of students in the Dartmouth College when they connect to the wireless access points of the campus.

4.1. Generating the synthetic data set

As stated above, we generate a synthetic data set of moving objects in supply chains. Similarly to (7), we consider several distribution centres or factories that may exchange tagged products/items in both directions by means of input/output gates (controlled by RFID readers). Once a distribution centre has M items in any of its output gates, it sends these items to another randomly selected distribution centre. Upon reception of a set of items by a distribution centre, these items are processed according to the distribution centre policy. Like in previous models (6) (7) (8), the distribution centre policy is defined by a graph. Locations where items arrive and depart are the nodes of the graph, whilst the edges represent the possibility of moving between locations. In particular, we define a random graph for each distribution centre and random Poisson distributions to model the departure of items in each location. By doing so, we simulate that items move in small groups or individually inside each distribution centre whilst they move in large groups between distribution centres. Note that this kind of movement is similar to the one given in (12) where two types of data are considered: (i) groups of items (GData) and (ii) single items (IData).

Similarly to (6), we define five distribution centres and twenty locations in each of them. For each distribution centre we define a random graph using an Erdős-Rényi model $G(n, p)$ where $n = 20$ and $p = 0.5$. Also, we assign to each location a Poisson distribution $P(\lambda)$ where $\lambda = 10$. Finally, the minimum number of items that are sent as a group between distribution centres is defined as $M = 100$.

In order to define the movement pattern of items we consider that they have different probabilities to departure towards different locations. For each out-edge of the graph, each item has a probability of taking this edge to leave. In our experiments, we have defined that for every node having n out-edges, the sequence of probability values assigned to these out-edges is a permutation of the sequence $\{\frac{1}{2}, \frac{1}{2^2}, \cdots, \frac{1}{2^{n-1}}, \frac{1}{2^{n-1}}\}$ (note that any other probability distribution could be defined.) Finally, considering all these settings, we generate a synthetic data set with 10^5 trajectories having an average length of 200 points.

4.2. Generating the real data set

Dartmouth College has 566 Cisco 802.11b access points installed to cover most of its campus. The college has about 190 buildings with 115 subnets so that clients roaming between buildings can change their IP addresses. This roaming information is recorded in different files for different clients by using syslog events (11). In total, more than $14,000$ trajectories collected over almost 2 years can be found in this data set.

For our experiments, we have selected the shortest $10,000$ trajectories of this data set. This subset of trajectories is created by parsing all the files having less than 46 Kb. We have selected the shortest trajectories because longer trajectories have useless, larger gaps in the data, generally caused by power failures, access points failures, or long periods of time in which clients were not in the campus. Note that, those big gaps should not appear in data sets of items moving through supply chains because, in this scenario, items cannot be considered lost for a long time. The trajectories of the resulting data set have an average length of 400 points.

4.3. Implementing predictors

In Section 3.5, we have defined an effective algorithm to predict the next location of moving objects based on a Markov model. Also, we have shown that it is possible to give an estimation of the time when an object should visit the next location.

In order to provide a better evaluation of our proposal, we have run experiments using two different predictors:

1. A Markov-based predictor: The predictor described in Section 3.5 used to estimate both the next location and the time when the object should visit that location.
2. An Oracle predictor: A predictor that always correctly guesses the next location and the time when the object should visit that location.

It should be emphasised that the Oracle predictor is only possible because we know in advance the trajectories of the data sets, otherwise it is not possible to create it. The Oracle predictor can be understood as the optimal predictor, *i.e.* an upper bound in prediction accuracy.

The Markov-based predictor that we have implemented for our experiments guesses correctly the next location and ETA of tags 41% of the the times with synthetic data, and 67% of the times with real data. The Oracle predictor has 100% of success for both data sets.

As it has previously been stated the performance of our protocol in terms of computational cost and bandwidth usage strongly depends on the accuracy of the predictors. Although the obtained results outperform all previous proposals, there is still room for improvement (*e.g.* by developing better predictors).

4.4. Protocols performance

We will compare the performance of the following proposals:

1. The Fouladgar *et al.* method (5) assuming that each tag is in the cache of only one reader. We refer to this method as **Fouladgar 1-1**.
2. The Fouladgar *et al.* method (5) assuming that each tag may be in the cache of several readers. The authors propose to store the data of tags in the cache of those readers that may read it most often. As this is not possible for the two data sets considered in this work, we make the assumption that a tag will be in the cache of the readers that have identified it previously. We refer to this method as **Fouladgar 1-M**.

3. The trajectory-based proposal of Trujillo and Solanas described in (20). We refer to this method as **T-S Trajectory Based**.
4. Our new proposal using a Markov-based predictor. We refer to this proposal as **Predictive (Markov)**.
5. Our new proposal using an Oracle predictor. We refer to this proposal as **Predictive (Oracle)**.

From a scalability point of view, the number of cryptographic operations performed on the server side is the main concern. Consequently, most of the hash-based protocols are not considered scalable. However, the performance of RFID protocols based on collaboration between readers lessen the computational cost of hash-based protocols but may be affected by an increase in the bandwidth usage. Therefore, for all the studied protocols we compute the number of cryptographic operations and, also, the number of messages sent amongst readers.

With the aim to study both, the computational cost and the bandwidth usage simultaneously, we have defined a trade-off measure that for every protocol outputs the percentage of *closeness* of the protocol to the optimal case; the higher *(closer)* the better.

Definition 5 (Trade-off measure)**.** *Let $\mathcal{P}$ be the set of protocols under evaluation. Let α be a real value in the range $[0..1]$. Let P_c and P_b be the number of cryptographic operations and the number of sent messages of a given protocol $P \in \mathcal{P}$. Let $min_c = \min(P^i_c),\ \forall\ P^i \in \mathcal{P}$, $min_b = \min(P^i_b),\ \forall\ P^i \in \mathcal{P}$, $max_c = \max(P^i_c),\ \forall\ P^i \in \mathcal{P}$, and $max_b = \max(P^i_c),\ \forall\ P^i \in \mathcal{P}$. Then, the trade-off measure that we propose is computed as follows:*

$$d(\alpha, P, \mathcal{P}) = \left(\frac{max_c - P_c}{max_c - min_c} \times 100\right) \times \alpha + \left(\frac{max_b - P_b}{max_b - min_b} \times 100\right) \times (1-\alpha)$$

The values of P_c and P_b are computed by simulating each protocol under the scenarios described in Sections 4.1 and 4.2. Using this measure, it is possible to globally analyse the performance of all protocols at the same time. In addition, thanks to the use of α, it is simple to weight the importance of either the computational cost or the bandwidth usage. Thus, it can be easily observed which of the analysed protocols perform best in given conditions.

4.4.1. Experiments over the synthetic data set

Figure 8 depicts the number of cryptographic operations performed by each protocol over the synthetic data set. In the beginning of the simu-

Figure 8: Average number of cryptographic operations performed by each protocol during the complete simulation with the synthetic data set (in red). The black line represents the moving average of those values in subsets of 100 elements. The time axis represents simulation steps.

Figure 9: The average number of cryptographic operations per identification for the simulation with the synthetic data set. (The lower the better)

Figure 10: Average number of messages sent by each protocol during the complete simulation with the synthetic data set (in red). The black line represents the moving average of those values in subsets of 100 elements. The time axis represents simulation steps.

Figure 11: The average number of messages sent per identification with the synthetic data set.

lation (in the start-up phase) the "Predictive (Oracle)" and the "Predictive (Markov)" have a performance similar to the "T-S Trajectory Based" protocol. However, after *learning* the movement pattern of items, they immediately outperform the "T-S Trajectory Based protocol". It can also be observed that the predictive protocols and the "T-S Trajectory Based" protocol are clearly superior to the "Fouladgar 1-1" and the "Fouladgar 1-M", thus confirming the results of (20). Figure 9 shows the average number of cryptographic operations per identification. From this figure, it is clear that our new proposals outperform the previous ones in terms of computational cost and, by extension, they improve scalability also.

Figure 10 shows the number of messages sent by readers in the studied protocols over the same data set, and Figure 11 depicts the number of those messages in average. It can be observed in both figures that the "Fouladgar 1-M" method sends fewer messages because it replicates the identification information of tags in several readers, at the cost of a poor scalability. It is also clear that our new proposals send a very similar number of messages to the "Fouladgar 1-M" proposal but they perform significantly better in terms of scalability.

Using the trade-off measure described in Definition 5 we have compared all the protocols considering different values of α (See Figure 12). It is apparent that the protocol presented in this article (in its two variants) is

Figure 12: The trade-off measure for each protocol and different values of $\alpha = \{0, 0.1, 0.2, \cdots, 1.0\}$. The higher the better. (Note that the colours assigned to the protocols do not coincide with the colours of the previous Figure).

Figure 13: Average number of cryptographic operations performed by each protocol during the complete simulation with the real data set (in red). The black line represents the moving average of those values in subsets of 1000 elements. The time axis is expressed in milliseconds.

Figure 14: Average number of cryptographic operations per identification with the real data set.

Figure 15: Average number of messages sent by each protocol during the complete simulation with the real data set (in red). The black line represents the moving average of those values in subsets of 1000 elements. The time axis is expressed in milliseconds.

Figure 16: Average number of messages sent per identification with the real data set

the best for almost all values of α. Only in the region of α values very close to 0 (meaning that only the number of messages counts) our proposal is not the best. Hence, we can conclude that our proposal is better than previous proposals for the analysed synthetic data set.

4.4.2. Experiments with the real data set

In the case of the real data set, we consider the same measures described above (*i.e.* the number of cryptographic operations, the number of sent messages, and the trade-off measure). Figure 13 and Figure 15 show the number of cryptographic operations and the number of sent messages for each protocol. Figure 14 and Figure 16 show those values on average. Finally, Figure 17 depicts the *closeness* of all protocols to the optimal case by using the trade-off measure described in Definition 5.

The results are very similar to the ones obtained with synthetic data. Again, our proposal outperforms all previous proposals. Note that the different shape of Figures 8 and 10 with respect to Figures 13 and 15 is due to the very nature of the analysed data (*i.e.* synthetic vs. real).

Figure 17: The trade-off measure for each protocol and different values of $\alpha = \{0, 0.1, 0.2, \cdots, 1.0\}$. The higher the better. (Note that the colours assigned to the protocols do not coincide with the colours of the previous Figure).

5. Conclusions and further Work

We have presented a novel protocol that allows the efficient identification of RFID tags by means of a set of collaborative readers. Our innovative proposal uses location and time of arrival predictors to improve the efficiency of the widely accepted IRHL scheme. We have shown that our protocol outperforms previous proposals in terms of scalability whilst guaranteeing the same level of privacy and security.

From the experimental results obtained, we can conclude that our proposal is comparable to highly scalable protocols like the tree-based protocols. However, we do not sacrifice any privacy to achieve this goal.

Usually, algorithms aimed at location prediction work well in some scenarios, but their performance decreases in others. Although we have provided some practical implementations for the predictors, the definition of our protocol is flexible enough to accept the use of any location predictor. Due to the fact that the efficiency of our proposal highly depends on the accuracy of the predictors, in the future, we plan to study and compare a variety of predictors in different scenarios.

Acknowledgements

This work was partly funded by the Spanish Government through project CONSOLIDER INGENIO 2010 CSD2007-0004 “ARES”, project TSI2007-65406-C03-01 “E-AEGIS”, project TIN2011-27076-C03-01 “CO-PRIVACY”, project INNPACTO 2010 PT-430000-2010-31, project AVANZA I+D 2009 TSI-020100-2009-720, project AVANZA I+D 2010 TSI-020302-2010-153, and by the Government of Catalonia under grant 2009 SGR 1135, and by the Rovira i Virgili University through project 2010R2B-02 “LOST?”.

The authors are with the UNESCO Chair in Data Privacy, but the views expressed in this paper are their own and do not commit UNESCO.

[1] Jordi Aragones-Vilella, Antoni Martinez-Balleste, and Agusti Solanas. A brief survey on RFID privacy and security. In Sio Iong Ao, Leonid Gelman, David W. L. Hukins, Andrew Hunter, and A. M. Korsunsky,

editors, *Proceedings of the World Congress on Engineering 2007*, volume 2 of *Lecture Notes in Engineering and Computer Science*, pages 1488–1493. Newswood Limited, July 2007.

[2] Gildas Avoine, Levente Buttyant, Tamas Holczer, and Istvan Vajda. Group-based private authentication. In *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2007)*, pages 1–6. IEEE, June 2007.

[3] Gildas Avoine, Etienne Dysli, and Philippe Oechslin. Reducing time complexity in RFID systems. In Bart Preneel and Stafford Tavares, editors, *Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 291–306. Springer Berlin / Heidelberg, 2006.

[4] Andrey Bogdanov, Lars R. Knudsen, Gregor Le, Christof Paar, Axel Poschmann, Matyhew J. B. Robshaw, Yannick Seurin and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In *9th International Workshop on Cryptographic Hardware and Embedded Systems*, CHES'07, pages 450–466. LNCS, 2007.

[5] Sepideh Fouladgar and Hossam Afifi. Scalable privacy protecting scheme through distributed RFID tag identification. In *Proceedings of the workshop on Applications of private and anonymous communications*, volume 3 of *AIPACa '08*, pages 1–8. ACM, 2008.

[6] Hector Gonzalez, Jiawei Han, Hong Cheng, Xiaolei Li, Diego Klabjan, and Tianyi Wu. Modeling massive RFID data sets: A gateway-based movement graph approach. *IEEE Transactions on Knowledge and Data Engineering*, 22(1):90 – 104, January 2010.

[7] Hector Gonzalez, Jiawei Han, Xiaolei Li, and Diego Klabjan. Warehousing and analyzing massive RFID data sets. In *Proceedings of the 22nd International Conference on Data Engineering*, ICDE '06, pages 83 – 92. IEEE Computer Society, April 2006.

[8] Hector Gonzalez, Jiawei Han, and Xuehua Shen. Cost-conscious cleaning of massive RFID data sets. In *ICDE 2007. IEEE 23rd International Conference on Data Engineering*, pages 1268 – 1272. IEEE, April 2007.

[9] Colin C. Haley. Are You Ready for RFID? *InternetNews.com*, November 2003. `http://www.internetnews.com/infra/print.php/3109501`.

[10] Ari Juels and Stephen A. Weis. Defining strong privacy for RFID. Reseach report 137, IACR e-print, April 2006. `http://eprint.iacr.org/2006/137`.

[11] David Kotz, Tristan Henderson, and Ilya Abyzov. CRAWDAD data set dartmouth/campus (v. 2005-03-08). Downloaded from `http://crawdad.cs.dartmouth.edu/dartmouth/campus`, March 2005.

[12] Chun-Hee Lee and Chin-Wan Chung. Efficient storage scheme and query processing for supply chain management using RFID. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, SIGMOD '08, pages 291–302. ACM, 2008.

[13] David Molnar and David Wagner. Privacy and security in library RFID: issues, practices, and architectures. In *Proceedings of the 11th ACM conference on Computer and communications security*, CCS'04, pages 210–219. ACM, October 2004.

[14] Adi Shamir. SQUASH - A New MAC with Provable Security Properties for Highly Constrained Devices Such as RFID Tags. In *15th International Workshop on Fast Software Encryption*, FSE'08, pages 144–157. LNCS, 2008.

[15] Agusti Solanas, Josep Domingo-Ferrer, Antoni Martinez-Balleste, and Vanesa Daza. A distributed architecture for scalable private RFID tag identification. *Computer Networks*, 51(9):2268–2279, 2007.

[16] Libo Song, David Kotz, Ravi Jain, and Xiaoning He. Evaluating location predictors with extensive Wi-Fi mobility data. In *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 1414 – 1424. IEEE, 2004.

[17] RFID Journal Staff. Air Canada GETS Asset Tracking. *RFID Journal*, March 2003. `http://www.rfidjournal.com/article/print/335`.

[18] RFID Journal Staff. Boeing finds the right stuff. *RFID Journal*, September 2003. `http://www.rfidjournal.com/article/print/715/`.

[19] Rolando Trujillo-Rasua and Agusti Solanas. Efficient probabilistic communication protocol for the private identification of RFID tags by means of collaborative readers. *Computer Networks*, 55(15):3211-3223, 2011.

[20] Rolando Trujillo-Rasua and Agusti Solanas. Scalable trajectory-based protocol for RFID tags identification. In *IEEE International Conference on RFID-Technology and Applications (RFID-TA 2011)*. IEEE, (to appear) 2011. `http://crises2-deim.urv.cat/docs/publications/conferences/633.pdf`.

[21] Bob Violino. Farm harvests RFID's benefits. *RFID Journal*, March 2004. `http://www.rfidjournal.com/article/print/810`.