RFID Readers Deployment for Scalable Identification of Private Tags

Agusti Solanas* **and Jesús Manjón**

Scalability is frequently used as a magic incantation to indicate that something is badly designed or broken. Often you hear in a discussion "but that doesn't scale" as the magical word to end an argument.

Werner Vogels
CTO - Amazon.com

Abstract The deployment of the RFID technology can put the privacy of its users in jeopardy. With the aim of averting the fears of the RFID potential users, a plethora of security and privacy methods have been designed. However, due to the important growth of this technology, scalability problems have arisen and the proper deployment of the technology has become a challenge.

In this chapter, we provide a brief overview of the most relevant methods for providing security and privacy to the users of the RFID technology. We pay a special attention to the hash locks proposal and we recall a method based on the distribution of RFID readers, with the aim to provide security and privacy in a scalable fashion.

In order to test our method, we have developed a simulator that is presented in this chapter. By using this simulator we study the distribution of several kinds of readers on a variety of scenarios, and we report some of the obtained results.

1 Introduction

Scalability is the property of a system, a network, or a process, which indicates its ability to deal with growing amounts of work in a proper manner. Monolithic systems (i.e. systems that rely on a single entity) generally suffer from scalability

A. Solanas
CRISES Research Group, UNESCO Chair in Data Privacy,
Department of Computer Engineering and Maths, Rovira i Virgili University,
Tarragona, Catalonia (Spain)
e-mail: agusti.solanas@urv.cat

P. Kitsos, Y. Zhang (eds.) *RFID Security: Techniques, Protocols and System-on-Chip Design,*

problems, and most of the times distributed systems (i.e. systems that consist of a number of connected entities) have been used to solve the limitations of the former ones. These limitations are related not only to scalability but also to a higher number of issues:

- *Fault-tolerance*. A monolithic system relies on a single entity (e.g. a mainframe). If this entity fails, the whole system collapses. On the contrary, the failure of an entity in a distributed system can be handled by the other entities. Although the service that the system provides (e.g. a web service) could be affected by the failure of an entity (e.g. a server), the system does not crash immediately and gives space for a reaction.
- *Price*. The Grosch's law says that computer performance increases as the square of the cost. If you want to do it twice as cheaply, you have to do it four times faster.[1] This law was originally applied to mainframes in 1965 but it has been seen that it does not hold with the new microprocessors technology. In fact, Grosch's law contradicts Moore's law, and it is quite clear that in terms of cost, it is better to increase the number of units instead of using powerful ones.
- *Incremental growing*. Although monolithic systems can increase their capabilities, they do not grow naturally. On the contrary, distributed systems have been designed to simplify the addition of new entities into the system. The ability of a distributed system of being easily enlarged is a clear advantage.
- *Bottlenecks*. Computer systems can be connected to a network in order to serve queries from users. Monolithic systems have to manage all the queries that arrive at a given time instant. When the number of queries grows, they are stored in a queue prior to being served. The use of monolithic systems may lead to an increase in the delays due to their lack of parallelism. On the contrary, distributed systems are able to balance the load amongst the elements of the system and delays are reduced.

Radio Frequency IDentification (RFID) technology is specially sensitive in terms of scalability because the number of RFID elements is growing faster with each passing day. RFID protocols must be secure and private, but at the same time, they have to be scalable if they want to be a reality. As we will see in Sect. 2, there are lots of proposals to manage the RFID systems securely and guaranteeing users privacy, but there is a lack of scalability. RFID devices have an important presence in our daily life, even when we do not see them, and they will become ubiquitous in the near future. The spectacular market push of RFID technology is due to the interest by large retailers (e.g. Wal-Mart[2]), important manufacturers (e.g. Gillette, Procter & Gamble, etc.) and governments. As a result, almost every object is liable to carry an RFID tag. RFID devices can be seen as a proper substitute of bar codes since they

[1] "There is a fundamental rule, which I modestly call Grosch's law, giving added economy only as the square root of the increase in speed – that is, to do a calculation ten times as cheaply you must do it 100 times as fast."

[2] Wal-Mart started to explore the RFID technology in 2003 and devoted at least three billion dollars to implement it [7].

are mainly used to identify objects. Unlike bar codes, RFID devices allow objects to be identified without visual contact and help in improving and automating many processes e.g. supermarket checkouts, product inventories, etc. This is possible due to the ability of RFID tags for being read fast and in parallel. An RFID system consists of two main components:

- *RFID tags*. They are small passive devices with a variety of possible appearances from stickers to small grains embedded in official documents. A tag basically consists of a microchip and a metal coil, which acts as an antenna. In some cases, it can also contain a battery and some other microchips intended for increasing its computational power.
- *RFID readers*. They are active devices used to read the information stored in the tags. In a nutshell, readers emit a radio wave so that all tags in their range *answer* by broadcasting their embedded information (i.e. a set of bits). This information, generally known as *Electronic Product Code* (EPC), is usually the identifier of the object into which the tags are stuck.

In order to see the relation between RFID systems and scalability and to illustrate our view, let us consider the next example:

Example 1. We want to design an application to control the vehicles crossing the borders of some western European countries. The idea is that each vehicle should be monitored in order to determine in which country it is. To do so, we assume that each vehicle has an RFID tag that identifies it. We also assume that a set of RFID readers have been distributed along the border in order to detect the tags in the vehicles. To tackle this problem, we can adopt a centralised (*monolithic*) or a distributed architecture. Figure 1 shows a graphical representation of both approaches.

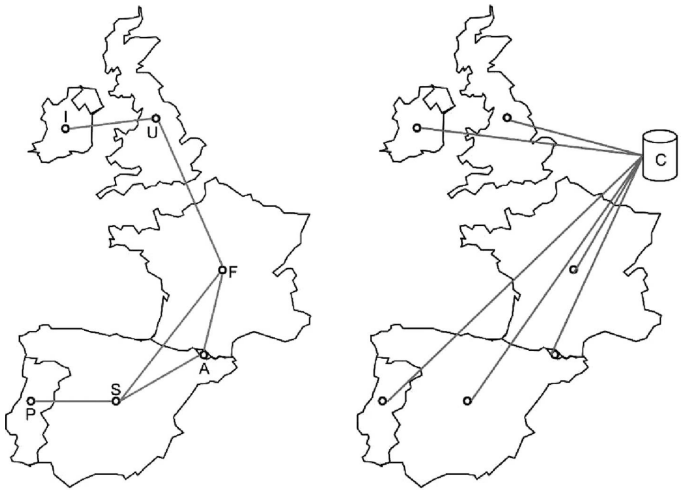

Fig. 1 Our border control example. *Left:* Adjacency graph for the distributed nodes of the example. *Right:* Centralised version of the example

In the centralised approach, all the border readers are connected to a central database C that contains all the possible IDs, let us say n IDs. Thus, when a vehicle crosses a border, a query to the centralised database is performed in order to determine whether the vehicle has permission to access the country. It is clear that this architecture does not scale because the delays will linearly grow as n grows and the number of queries that C has to manage will linearly grow as n grows also.

On the contrary, the distributed approach considers an adjacency graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where the vertexes $\mathcal{V} = \{P, S, A, F, U, I\}$ represent the distributed databases located in each country, and the edges $\mathcal{E} = \{(P \leftrightarrow S), (S \leftrightarrow A), (S \leftrightarrow F), (A \leftrightarrow F), (F \leftrightarrow U), (U \leftrightarrow I)\}$ represent the possible ways that a vehicle can take. Note that the distributed databases do not store n IDs. Each database stores the IDs of the vehicles which are in its influence area (i.e. it stores the IDs of the vehicles in its country and its adjacent countries). It is not necessary to store the IDs of the vehicles in I in the database in P, because it is impossible that a vehicle from I reaches P without crossing S. By applying this distributed strategy, the number of IDs that must be stored in each database is clearly lower that n and the system scales.

1.1 Contribution and Plan of the Chapter

In this chapter, we study one of the most important problems that the RFID technology has to face i.e. scalability. We consider the scalability problem of the RFID technology from a security and privacy point of view. We want to obtain secure and private RFID systems able to scale properly with the growing number of tags.

With the aim being to define a reference framework, in Sect. 2 we provide a brief overview of the private and secure RFID protocols proposed in the literature. Next in Sect. 3, we deeply analyse the hash locks approach and, we show how its scalability problems can be overcome by means of the distribution of RFID readers and a communication protocol amongst these readers. Section 4 presents our simulator. We show the main capabilities of the simulator and we explain how to use it. In Sect. 5 we report a number of experimental results that we have obtained using our simulator. We study a variety of scenarios and we show that the protocol that we propose scales properly. Finally, in Sect. 6 the chapter concludes by pointing out some future challenges.

2 Brief Comments on RFID Security and Privacy

There is a variety of methods for providing RFID tags with privacy. The proposed methods mainly depend on the computational power of the tags in which they will be run. In general, we can divide the methods considering the computational constraints of the tags:

1. *Elemental or basic tags*, which are not capable of performing cryptographic operations such as generating random values or computing hashes. This kind of tags can be disabled by using killing or sleeping commands. A different approach is the use of proxies [5, 18], but they require the users to have some technical background. The blocking approach [15] is another method that relies on the concept of *blocker tag* that simulates the full spectrum of possible tags. By doing this, it becomes very difficult for a reader to know which tags are really being carried by a given user. Sarma, Weis and Engels [19] proposed the re-labelling of tags in order to avert their tracking. Innoue and Yasuura [10] and Good et al. [6] proposed some variants of this approach. The use of pseudonyms has been also proposed [11] and re-encryption has been also considered [13].
2. *Symmetric-key tags*, which are capable of dealing with symmetric-key cryptography protocols. These tags can deal with more complex protocols like the OSK by Ohkubo, Suzuki and Kinoshita (OSK) [16], and some of its variants [1, 8]. The YA-TRAP protocol by Tsudik [22] is another candidate for being used with this kind of tags. Weis, Sarma, Rivest and Engels proposed the use of hash locks and presented the *deterministic hash lock* approach in [24], later Juels proposed an improvement called *improved randomised hash locks* in [12]. Henrici and Müller in 2004 [9] also proposed a privacy protocol for this kind of tags and Engberg, Harning and Jensen [4] provided an alternative based on zero-knowledge protocols.
3. *Public-key tags*, which are capable of managing public-key cryptography protocols. Some protocols have been proposed for this kind of tags based on Elliptic Curve Cryptography (ECC) [2], the Elliptic Curves Discrete Logarithmic Problem (ECDLP) [17], or Physical Unclonable Functions (PUF) [3, 23].

Symmetric-key tags are very well suited for most real-world applications, specifically the hash locks approach is a very interesting method for providing privacy to the RFID technology. We will briefly introduce the main idea of deterministic hash locks in Sect. 2.1. Next, we will focus on the improved randomised hash locks in Sect. 2.2.

2.1 Deterministic Hash Locks

Weis, Sarma, Rivest and Engels proposed in 2003 the use of hash locks in RFID devices. A first approach, called *deterministic hash locks*, was presented in [24]. A tag is usually in a "locked" state until it is queried by a reader with a specific temporary meta-identifier Id. This is the result of hashing a random value (*nonce*) selected by the reader and stored into the tag. The reader stores the Id and the *nonce* in order to be able to interact with the tag. The reader can unlock a tag by sending the *nonce* value. When a tag receives it, the value is checked. Another way of running this scheme is by using some *meta-keys*. Each tag is initialised with a $(Id, \text{meta-key})$ pair, then, in order to unlock the tag, the *meta-key* is used. The problem of this

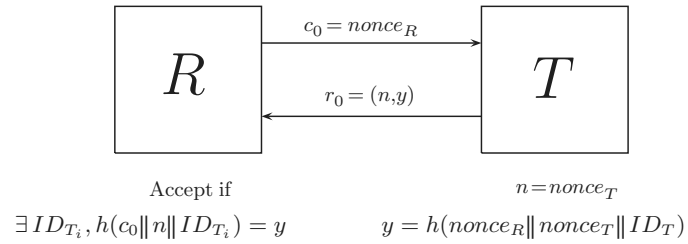

Fig. 2 Diagram of Juels–Weis improved randomised hash locks

solution is the cost of storing these pairs. Note that the hash approach does not suffer from this shortcoming. Another security problem that must be faced is how to securely send the meta-identifiers from readers to tags and vice versa.

2.2 Improved Randomised Hash Locks

A recent approach based on hash locks can be found in [12], where the *improved randomised hash locks* are presented. The basic operation of the improved randomised hash locks is depicted in Fig. 2 and is next briefly described:

1. A reader R sends a challenge c_0 to a tag T, where $c_0 = \text{nonce}_\text{R}$ is generated uniformly at random.
2. T generates its own nonce nonce_T and hides its unique identifier ID_T by sending a response $r_0 = (\text{nonce}_\text{T}, h(\text{nonce}_\text{R}||\text{nonce}_\text{T}||\text{ID}_\text{T}))$.
3. To determine ID_T, R must perform an exhaustive search of the IDs in its database to compute $r_i = (\text{nonce}_\text{T}, h(\text{nonce}_\text{R}||\text{nonce}_\text{T}||\text{ID}_{\text{T}_i}))$ and compare the result with r_0. Once R finds an ID_{T_i} that satisfies $r_i = r_0$, the tag is identified.

In [14] it is proved that improved randomised hash locks offer strong tag privacy in front of eavesdroppers. The main limitation of this technique is scalability: indeed, the authors of [14] express their belief that, for RFID tags capable of only symmetric-key cryptography, their definition of strong privacy may require the reader to perform brute-force search to identify tags, which scales poorly. They also point out the need of definitions and protocols for RFID privacy that are weaker, but more *practical* and *useful*. We absolutely agree with their remarks and we concentrate on the definition of a protocol permitting the practical use of privacy schemes like the improved randomised hash locks in a scalable manner.

3 Our Approach to Scalability

With the aim being to provide scalability to hash-locks-like protocols, we recall a solution that can be divided in two main parts [20]:

- *A readers deployment*. Readers must be deployed properly in order to cover the tags movement area.
- *A protocol suite*. The protocol is used by the readers to exchange information on the location of the tags. By exchanging this information, the readers do not need to store huge amounts of tag IDs, but only the ones that are in their cover range or close to it.

In Sect. 3.1, we explain how to distribute the readers in order to let them collaborate to control the tags flow. Next, in Sect. 3.2 we detail our protocol.

3.1 Readers Deployment

We now describe the spatial distribution of readers and the distribution of IDs amongst them. Consider an area Ψ that can be covered by a number of readers. Assume that tags enter and leave Ψ through designated points called system access points (SAP) and system exit points (SEP), respectively.

Readers are placed according to a grid pattern depicted in Fig. 3. Let A_i be the square cell covered by the ith reader R_i. For the sake of clarity, we consider that all readers have the same cover range, so that all cells have the same size. However, our protocol has been designed to deal with variable cover range readers as we will show in Sects. 4 and 5. Further, we consider cells to be disjoint and to span the entire area Ψ. Formally,

$$\bigcup A_i = \Psi, \forall i. \tag{1}$$

$$A_i \cap A_j = \emptyset, \forall i, j | i \neq j. \tag{2}$$

It is assumed that the readers are able to locate a tag by collaborating. Although the technologies for locating a tag are beyond the scope of this work, we next discuss some relevant issues about tag location.

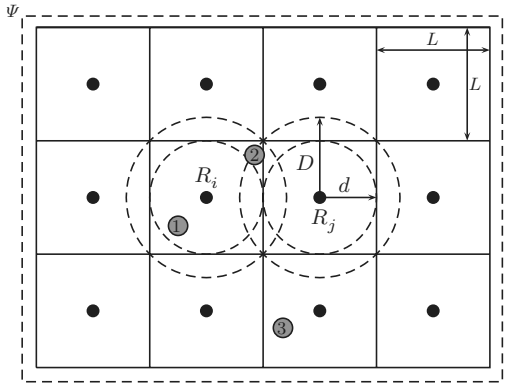

Fig. 3 Scheme of the coverage of a set of readers. The *numbers* are used to indicate different tag location situations [20]

Let D be the radius of the smallest circle containing a square cell and d the radius of the greatest circle inscribed in a square cell. Depending on the location of the tag, readers can face three different situations (see numbers in Fig. 3):

1. The distance between the tag and the reader R_i is less than d: In this case, the tag is located in the square area covered by R_i.
2. The distance between the tag and R_i is less than D and greater than d: In this case R_i needs the help of an adjacent reader R_j to determine the location of the tag. Note that it is not necessary to exactly locate the tag: determining its current cell is enough. Thus, it is possible to fulfil this task with only two readers.[3]
3. The distance between the tag and R_i is greater than D: In this case, the tag is off range of R_i.

3.2 Information Sharing Protocol Suite

Our method is based on sharing information between readers. The shared information is the tag ID and the ID of the reader which covers the cell in which a certain tag is located. The readers use three kinds of messages to share information: tag arrival, tag roaming and tag departure (*cf.* Table 1). This information is stored in the local cache of each reader involved in the message exchange.

In order for information to be shared without unnecessary replication, each reader removes from its cache the information related to tags which are no longer in the reader's cover range or in the cover range of an adjacent reader.

The suite consists of three protocols corresponding to the life cycle of a tag with respect to the system (i.e. arrival, roaming and departure):

1. *Arrival protocol*. It starts when a new tag enters the system. Upon arrival of a tag, a number of messages are sent in order to propagate the tag ID and the ID of the reader that acted as a SAP.
2. *Roaming protocol*. It is used when a tag moves from the cover range of a reader to the cover range of another reader. As a result, a number of ID propagation messages are generated and a request of ID deletion is also sent to the readers which are no longer accessible by the tag from its new location.

Table 1 Message types used by the information sharing protocol suite

Message	Meaning
$(T, \oplus)$	Tag T enters into the system
(T, R_i)	Tag T enters into the cover range of reader R_i
$(T, \ominus)$	Tag T leaves the system

[3] If the exact location was needed, at least three readers ought to be used.

3. *Departure protocol*. It is responsible for managing the departure of tags from the system. It generates a number of ID deletion notifications to rid the readers of the IDs of the departed tags.

3.2.1 Arrival Protocol

Let system access points (SAP)s be the entrances to the system. The number of SAPs and their locations is variable and user-definable, i.e. they depend on the nature and lay-out of the facility (airport, factory, store, etc.) served by the RFID system.

Each SAP is supposed to know all the possible tags which can enter the system through it. For example in a wholesale distribution centre, fresh fish enters through gate 1 served by SAP1, cleaning products enter through gate 2 served by SAP2, etc. Thus, SAP1 only needs to know the information about fresh fish and SAP2 only needs to know the information about cleaning products. Thanks to this division, the amount of information stored in the SAPs scales better and goods can enter the system in an orderly fashion.

We assume that a SAP consists of a reader connected to a computer that can efficiently access a database of tag IDs. Regarding the remaining readers (those which are not SAPs), they can be very simple devices with little storage and computational capabilities.

Note 1. This approach substantially differs from a centralised scheme in which all readers are connected to a back-end computing system. In our approach, only SAPs need a connection to the back-end and only the incoming tags are considered, which increases scalability.

For the sake of simplicity we describe the protocol with a single SAP and a single reader R_{in}. The generalisation to multiple SAPs is straightforward.

The arrival protocol is as follows:

Protocol 1 (Tag Arrival)

1. *The protocol starts when a tag T is detected by a SAP (cf. Step 1 of Fig. 4). If T is not found in the SAP database then the SAP raises an alarm to inform that there is an unidentified tag trying to enter the system without permission.*

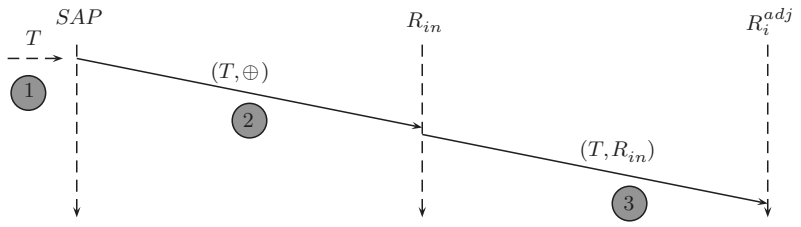

Fig. 4 Messages generated upon the arrival of an authenticated tag into the system

2. *If the tag is correctly identified, the SAP sends a message to* R_{in} *in order to inform the reader that a new tag* T *is going to enter the reader's cover range (cf. Step 2 of Fig. 4).*
3. R_{in} *adds the tag* T *to its cache; thus, when* T *reaches the cover range of* R_{in}*, the reader is able to authenticate* T*. After authentication,* R_{in} *sends a message to all its adjacent readers* $R_{\text{in}}^{\text{adj}}$ *to inform them that* T *has entered* R_{in}*'s cover range and can roam to any of them (cf. Step 3 of Fig. 4).*
4. *In response to that message, the adjacent readers* $R_{\text{in}}^{\text{adj}}$ *add* T *to their caches and record the name of the message originator (i.e. in this case* R_{in}*) who is the current owner of the tag.*

3.2.2 Roaming Protocol

The roaming protocol task is to share the tag ID information needed for updating the caches of the readers. By properly updating their caches, the readers can authenticate the tags in their cover ranges, and they can also leverage their resources (memory allocation, computational power). To that end, the ID information of the tags controlled by a given reader must be shared with adjacent readers, and non-adjacent readers must remove the ID information from their caches. Such a removal averts the uncontrolled growth of the readers' caches and makes the system scalable in terms of computational cost and memory space.

The roaming protocol is launched when any tag T moves from the cover range of a reader to another (adjacent) reader. The protocol is as follows:

Protocol 2 (Roaming)

1. *A tag* T *is detected by a reader* R_i *other than the owner of the tag (cf. Step 1 of Fig. 5), where we denote by owner of* T *the last reader that informed the rest of readers that* T *was in its cover range. Due to the spatial distribution of the readers,* T *must come from one of the adjacent readers to* R_i*, so* R_i *has in its cache the ID information of* T *and is able to identify it.*
2. *After identification,* R_i *sends a message to its adjacent readers* R_i^{adj} *in order to inform them that the new owner of* T *is* R_i *(cf. Step 2 of Fig. 5).*
3. *Upon message reception, the adjacent readers behave differently depending on their current cache information:*

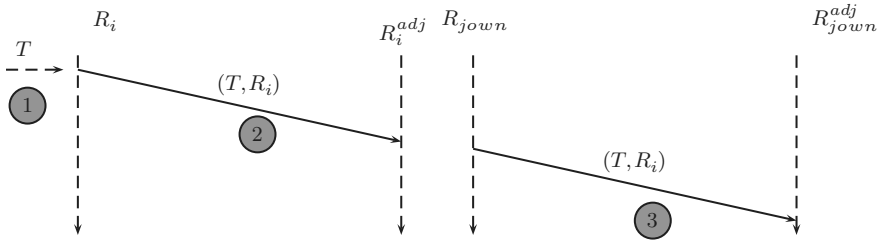

Fig. 5 Messages generated during the roaming protocol

(a) If an adjacent reader $R_j \in R_i^{\text{adj}}$ has no information about T *in its cache then it simply appends T and its owner information.*

(b) If an adjacent reader $R_j \in R_i^{\text{adj}}$ has information about T in its cache but was not the previous owner of T, then it only needs to update the ID of the owner of T (i.e. in this case R_i).

(c) The adjacent reader R_{own} which was the previous owner of T must communicate to its adjacent readers $R_{\text{own}}^{\text{adj}}$ that the new owner of T is R_i. To do so, R_{own} propagates the message from R_i to their adjacent readers (cf. Step 3 of Fig. 5).

4. *When the adjacent readers of R_{own} receive the message, they behave differently depending on their adjacency relations:*

(a) If a reader $R_k \in R_{\text{own}}^{\text{adj}}$ is adjacent to R_i then it does nothing.

(b) If a reader $R_k \in R_{\text{own}}^{\text{adj}}$ is not adjacent to R_i then it removes the information on T from its cache.

5. *At the end of the protocol, only readers adjacent to the current owner keep information on T in their cache.*

3.2.3 Departure Protocol

In almost any RFID application, tags which have entered a controlled system must leave it. For example, in a supermarket, grocery, warehouse, etc., tags travel from the shelves to the checkout.

In order to control the departure of the tags from the system, the SEPs are used. A SEP is an area covered by a reader from which no tag can go back into the system (e.g. a checkout in a supermarket).

The departure protocol works as follows:

Protocol 3 (Departure)

1. *The protocol starts when a tag T is detected by a SEP (cf. Step 1 of Fig. 6).*
2. *The SEP informs its adjacent readers $R_{\text{SEP}}^{\text{adj}}$ that T must be removed from their caches because there is no chance for T to go back (cf. Step 2 of Fig. 6). The adjacent readers of the SEP, including the previous owner R_{own} of T, erase the information on T from their caches.*

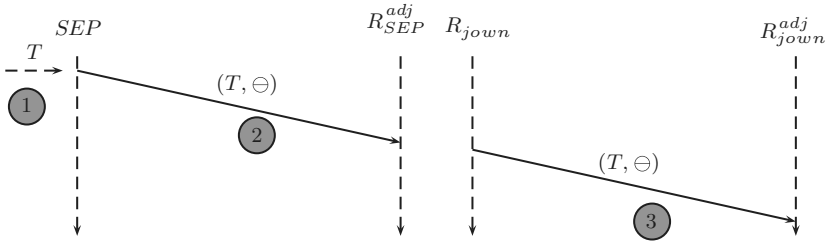

Fig. 6 Messages generated during the departure protocol

3. *The previous owner* R_{own} *of* T *propagates the removal message to its adjacent readers* $R_{\text{own}}^{\text{adj}}$ *(cf. Step 3 of Fig. 6).*
4. *The readers* $R_{\text{own}}^{\text{adj}}$ *remove any information on* T *from their caches, and no data remain in the system about the departed tag.*

4 Our Simulator

After the massive irruption of the RFID technology in the retail sector, many RFID manufacturers have developed a number of simulators to analyse the efficiency and reliability of their products. However, most of these simulators are basically centred in the study of a specific RFID device. Although these simulators are useful to determine the capabilities of a given device, they are useless for analysing the scalability of a whole RFID system. Moreover, it is difficult to find free open-source simulators and, more complex suites such as *iAnywhere* [21] are expensive commercial applications. Thus, one of the greatest problems when measuring the efficiency of a new protocol is the lack of tools.

With the aim being to overcome this limitation, we have implemented an RFID simulator that is able to analyse the scalability of a given protocol. Our simulator has been implemented in ANSI C and the graphical user interface (GUI) uses the GTK 2.0 libraries. Thus, the simulator is portable to different platforms such as Windows, Linux and Mac.[4]

Figure 7 shows the main parts/modules of the simulator. In the next sections, we provide the reader with a general idea on how the main modules of the simulator

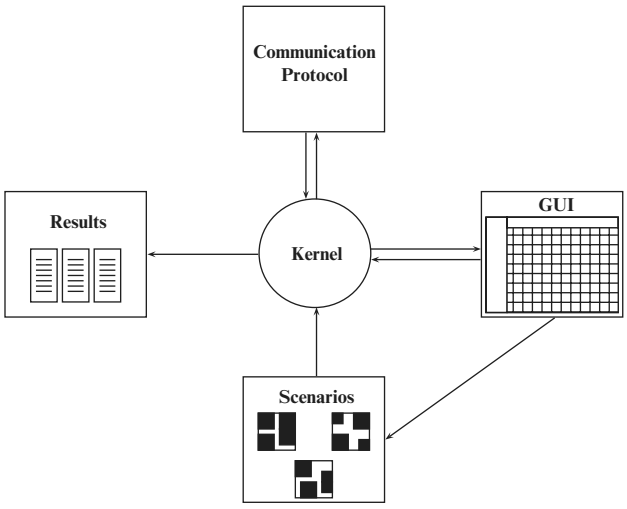

Fig. 7 Main modules of our RFID simulator

[4] Currently the simulator has versions in Windows and Linux.

work. Note that we will not go into details on how the modules have been implemented but we only give a conceptual approach.

4.1 The Kernel of the Simulator

The kernel is a central module of the simulator, it controls the flow of tags entering into and leaving from the system and interacts with the communication protocol module to simulate the readers network.

The entrance of tags into the system is controlled by two main parameters: the maximum number of tags and the frequency of arrival. The movement of the tags is also controlled by this module (in this version of the simulator the movement of the tags is random).

The kernel collects all the data and stores them in the results files. It also controls the number of tags in the system and stops the simulation once all the tags have left the scenario.

4.2 The Communication Protocol Module

This module defines the behaviour of the simulated readers and their interaction. It is an event-oriented module in the sense that it reacts to the events that the kernel raises. When a tag changes its location, the kernel raises a tag movement event and the communication protocol module must catch the event and update the state of the readers.

In this chapter we have considered the protocol described in Sect. 3. However, the simulator is able to deal with virtually any protocol of this kind.

4.3 Scenarios Management

An interesting feature of our simulator is the possibility to design and manage different scenarios. In order to study the scalability of a protocol completely, it is necessary to test it in different situations and observe its behaviour in bottlenecks, big areas without obstacles, narrow corridors, etc. By using our simulator it is simple to build new scenarios with a variety of features. (see Table 2)

Initially, a scenario is an empty space divided into a number of squares (e.g. 20×30 squares) that form a grid. A square is the smallest unit that we consider in the simulation (i.e. each tag is located in a square at a given time). The number of squares in each scenario can be defined by the user. Once the total number of squares is defined, the user can place SAPs and SEPs, simulating the doors that would exist in a real environment. Then, the readers are distributed amongst the squares and their cover ranges are configured.

All the scenarios can be saved and reused as times as needed, and they are easily exportable to applications such as Excel or Matlab.

4.4 The Graphical User Interface

The graphical user interface (GUI) allows the users to interact with the simulator. The GUI has three main objectives:

1. *Help the user in building and managing scenarios*. During the definition of scenarios, a crucial point is the placement of the readers and their configuration. The simulator allows to modify the cover range of each reader in a simple way and gives the possibility of studying the behaviour of a protocol on the same scenario but with different configurations of the readers. In Fig. 8 we can observe that each reader (marked with a black circle) is configured with a given cover range (e.g. at the corridors located on the left of the scenario, readers have a cover range of a single square, while in the room located on the right the cover range of the readers is 25 squares).

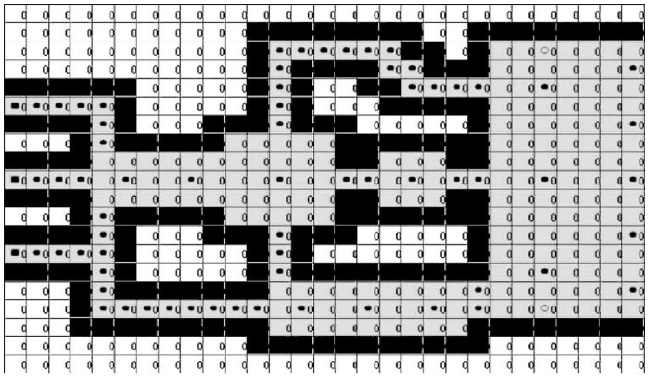

Fig. 8 A view of an scenario with different readers

Table 2 Symbols used to create scenarios

Symbol	Meaning
	Reader covering a cell
	Wall
	System Access Point (SAP)
	System Exit Point (SEP)

2. *Allow the user to monitor the evolution of a simulation*. If desired, the user can monitor the evolution of the simulation. The movement of the tags and the reader's cache size are shown at each simulation step.
3. *Permit the user to define the desired outputs*. As we show in the outputs module section, there is a number of results that can be obtained from the simulation and the user is able to select which one he/she wants to get.

4.5 The Outputs Module

During a simulation, information about the number of tags in each cell and the state of the caches of the readers is collected. Files are created at different steps of the simulation and they later allow an accurate study of the evolution of the caches of the readers over time.

In this version of the simulator, we have mainly focused on the size of the caches of the readers. However, it is easy to extend the simulator to let it gather a variety of information related to the simulated protocol.

4.6 Some Additional Features

In addition to the described features, the simulator has other complementary ones to make easy the study of the results of the simulation.

As shown in Fig. 9, it is possible to plot the path that a particular tag has followed. This path is seen graphically while the simulation takes place, and it can be checked after the simulation. Before the simulation starts, it can be checked whether a scenario is correct. A scenario is considered to be correct when it has access and exit points, it is surrounded by walls (i.e. it is finite), and it is totally covered by readers. If a scenario is not correct the simulation will not start and the program will

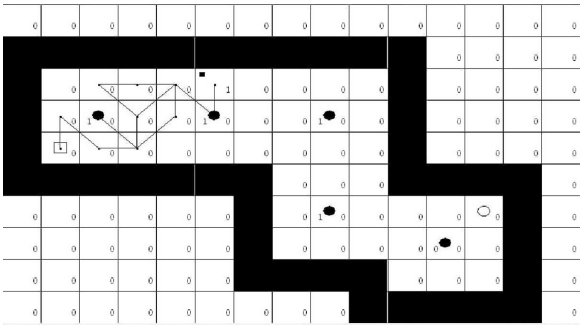

Fig. 9 Example of the path of a tag

warn the user about the error. Finally, the simulator allows to observe the connections amongst the readers. In our scalability protocol, these connections are essential since the readers exchange information with all their neighbours.

5 Experimental Results

In this section, we report a number of experiments that we have carried out by using our simulator. We have created five scenarios which have interesting features such as corridors, bottlenecks, small and big rooms, etc. We have configured the readers to have different cover ranges, thus, we can study the load that the most powerful readers have to manage. Considering a random movement of the tags, we have mainly concentrated on the analysis of the size of the caches of the readers.

In Sect. 5.1, we recall our border control example explained in the first section, and we show how to simulate it by using our simulator. Next, in Sect. 5.2 we analyse the proposed scenarios in detail.

5.1 Our Example

The simulation of our example is very easy and it will help us to clarify how a simulation is done. First, the scenario is built by using the GUI and the readers, the SAPs and the SEPs are placed in it. Figure 10 shows the scenario that represents our border example depicted in Fig. 1. We have configured all the readers to have the same cover range. By doing this, we guarantee that the load of each simulated local database is similar (i.e. we assume that the number of vehicles does not depend on the surface of the country[5]). We have placed two SAPs, one in *P* and the other in *I*, and a SEP in *A*. These points are placed arbitrarily and they are intended for an illustration purpose only.

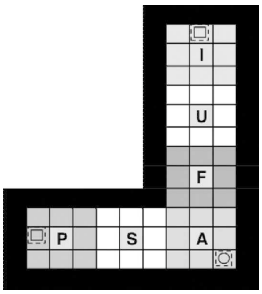

Fig. 10 Example represented in the simulator

[5] This assumption is unrealistic, but it is good enough for our example.

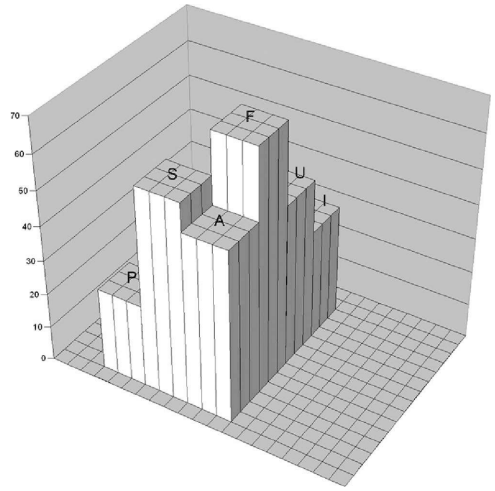

Fig. 11 Results of the simulation of the example

Once the scenario is built, we start a simulation with a few tags (e.g. 500 tags) and we obtain some results on the size of the caches of the readers. Figure 11 shows an example of the obtained results when simulating our example. From the study of these results we can emphasise some conclusions:

1. Although A is connected to very loaded nodes (i.e. S and F), it has a lower load because the SEP is located in its cover range.
2. P and I have a low load because they are only connected to another node.
3. S and F are the most loaded readers because they are connected to three other readers, thus, they have to store in their caches the information of all these readers.

5.2 Some Other Scenarios

In [20], we studied the scalability of the protocol described in Sect. 3 considering that all the readers had the same cover range (i.e. one square). Although this study was useful to show that the scalability of the RFIDs system is clearly improved, it is necessary to consider more complex scenarios in which the cover range of the readers is diverse. Moreover, it is also interesting to determine the influence of the cover range on the scalability of the system. To do so, we have designed five scenarios (*cf.* Fig. 12) and we have covered them with a variety of readers. The scenarios that we have built can be summarised as follows:

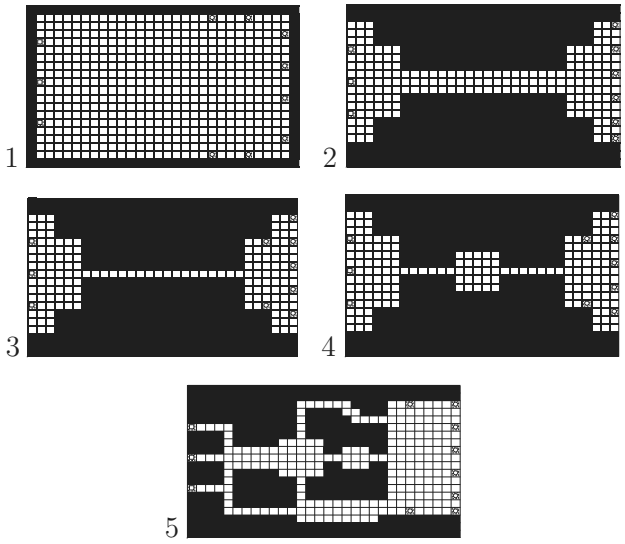

Fig. 12 Scenarios used in our simulation study

1. *Empty scenario*. This scenario has three SAPs and eight SEPs, there are no obstacles or walls in it, thus, it is useful to show the influence of the readers' cover range on the scalability of the system.
2. *Scenario with a wide corridor*. This scenario has three SAPs and six SEPs. By using this scenario, we analyse how the corridors and their associated bottlenecks affect the scalability of the system. Moreover, we consider different options for covering the area with heterogeneous readers.
3. *Scenario with a narrow corridor*. This scenario has three SAPs and seven SEPs. It is a scenario similar to the previous one, but in this case the corridor is narrow. Thus, the corridor can only be covered by one-square range readers.
4. *Scenario with a narrow corridor and a small room in the middle*. This scenario has three SAPs and seven SEPs. In this case we can study the bottlenecks that appear at the entrance and the exit of the small room in the middle of the corridor.
5. *Complex scenario with corridors and small and big rooms*. This scenario has three SAPs and eight SEPs. It is the most realistic of all the analysed scenarios. It has been designed to study how the protocol behaves in a real and complex scenario.

5.2.1 Empty Scenario Analysis

This scenario has 600 squares which have been covered by readers having a cover range between 1 and 25 squares. Figure 13 shows the size of the caches for different cover ranges when the maximum number of tags into the system is 1,000. It can be

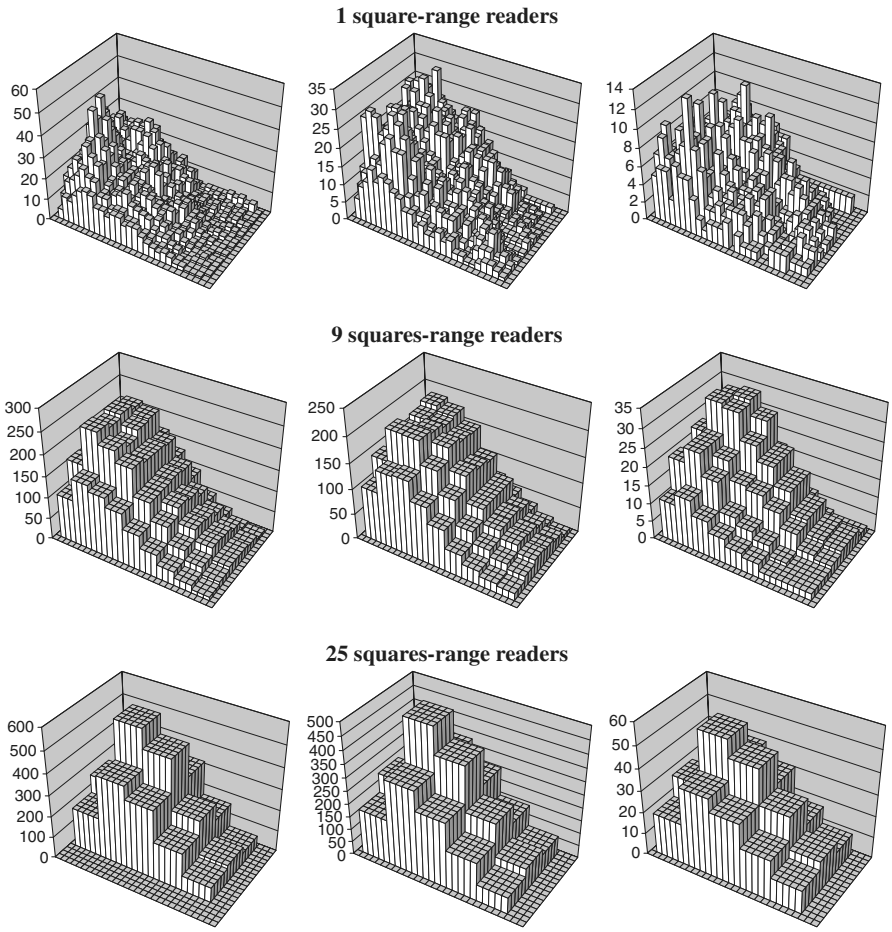

Fig. 13 Evolution over time (from *left* to *right*) of the caches of the readers in the first scenario covered by 1, 9 and 25 squares-range readers with a maximum of 1,000 tags in the system

seen that most of the tags are located close to the SAPs. This happens because the tags move randomly and there are no obstacles in the scenario. Moreover, once a tag reaches a SEP it leaves the system. Hence, the density of tags which are close to a SEP is very low. The average cache size of the readers grows with the number of covered squares. Specifically, when the cover range is 1 square, the average is 13.9, when the cover range is 9 squares, the average is 116.8 and, when the cover range is 25 squares, the average is 261.7. In all cases the cache size is much lower than the total number of tags (i.e. 1,000 tags). The same behaviour can be observed when the number of tags grows to 10,000 and 50,000 (*cf.* Figs. 14 and 15).

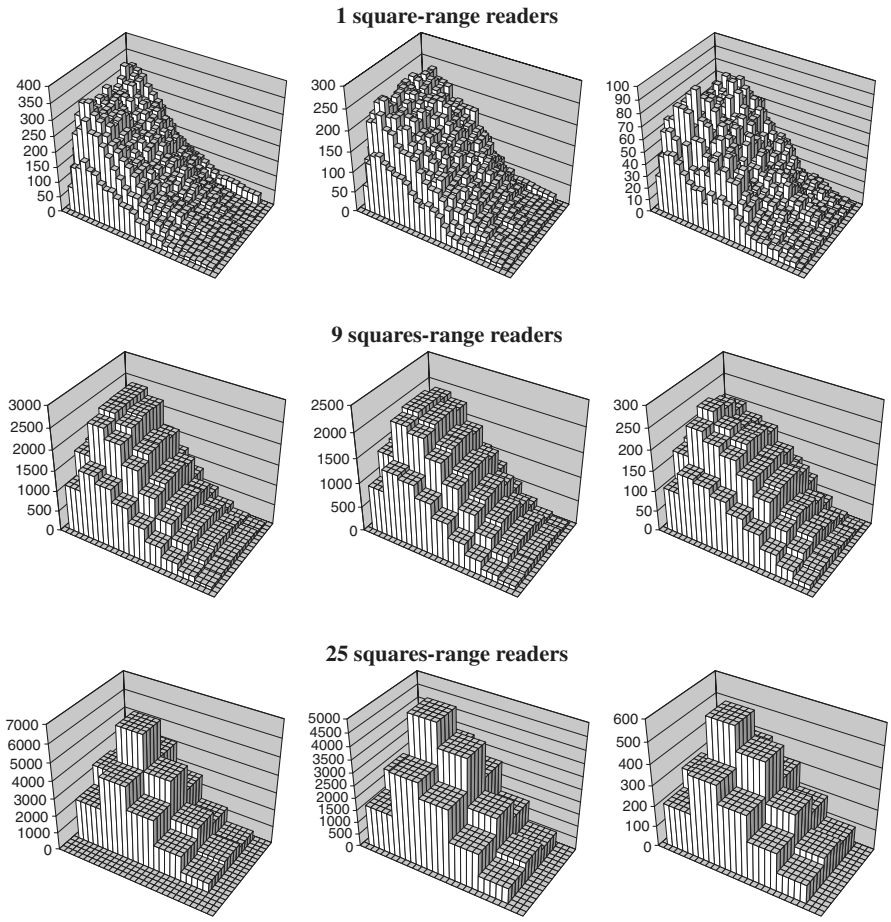

Fig. 14 Evolution over time (from **left** to **right**) of the caches of the readers in the first scenario covered by 1, 9 and 25 squares-range readers with a maximum of 10,000 tags in the system

5.2.2 Analysis of Scenarios with Corridors

In this scenarios (i.e. 2–4) we are specially interested in analysing the bottlenecks which appear at the entrance and at the exit points of the corridor, and how the cover range of the readers affects their cache size. To do so we have introduced 10,000 tags into the system and, as Fig. 16 shows, the behaviour of the caches follows a similar pattern than in the previous scenario. Most of the tags are located at the entrance of the system near to the SAPs. On the other hand, the density in the corridor is very uniform and the only *dangerous* point is the entrance to the corridor. If we analyse the average size of the caches we observe that in the second scenario with 1 square cover range readers the average is 323, if we use 9 squares cover range readers the average is 1,842.9 and 2,265 when 25 squares cover range readers are used. In the

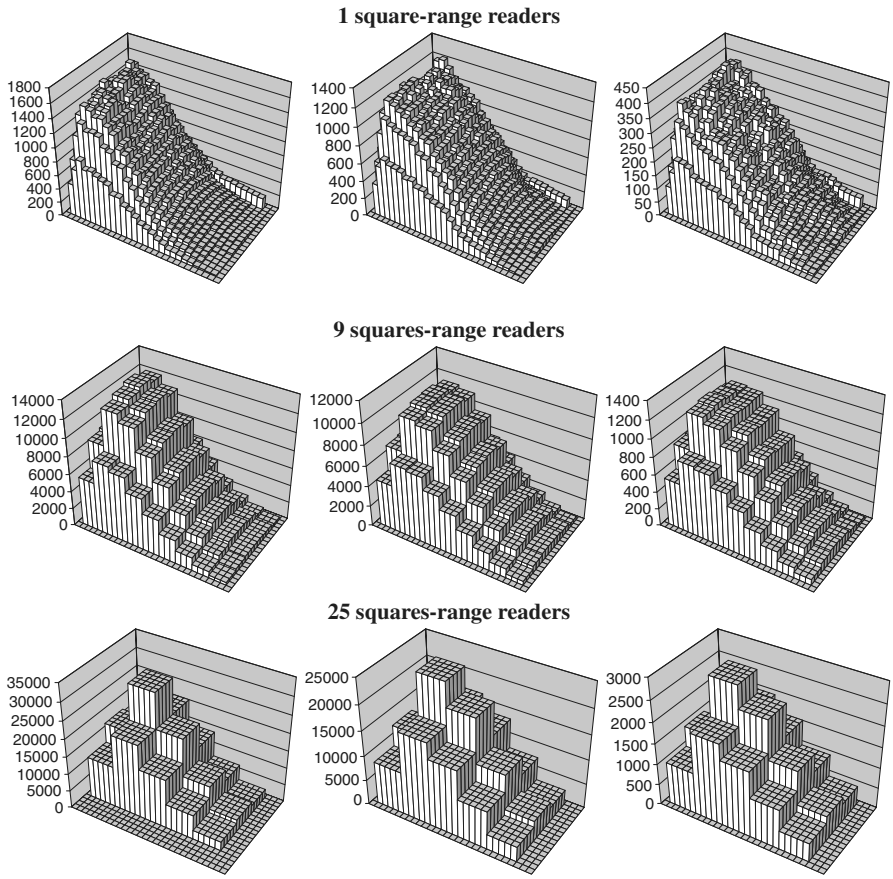

Fig. 15 Evolution over time (from *left* to *right*) of the caches of the readers in the first scenario covered by 1, 9 and 25 squares-range readers with a maximum of 50,000 tags in the system

third and fourth scenario, where a mixture of readers has been used, the average size of the caches is 2,282.7 and 1,064.2, respectively (Figs. 17 and 18).

The most relevant result that must be emphasised in these scenarios is that the use of 25 squares cover range readers does not increase substantially the average size of the caches, thus, it seems that the use of higher range readers in bottlenecks should be a good choice.

5.2.3 Complex Scenario Analysis

This scenario has been designed to show that the mixture of different cover range readers scales properly in a complex environment full of corridors, bottlenecks and big and small rooms. Figure 19 shows the cache sizes of the readers used in this

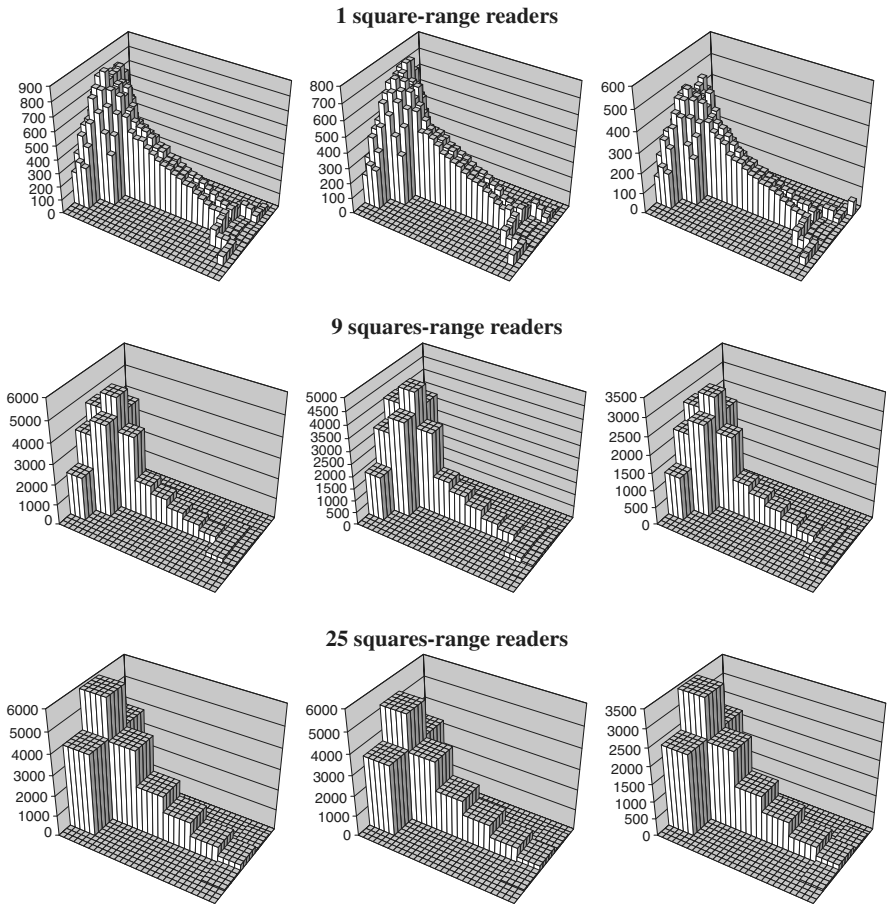

Fig. 16 Evolution over time (from *left* to *right*) of the caches of the readers in the second scenario covered by 1, 9 and 25 squares-range readers with a maximum of 10,000 tags in the system

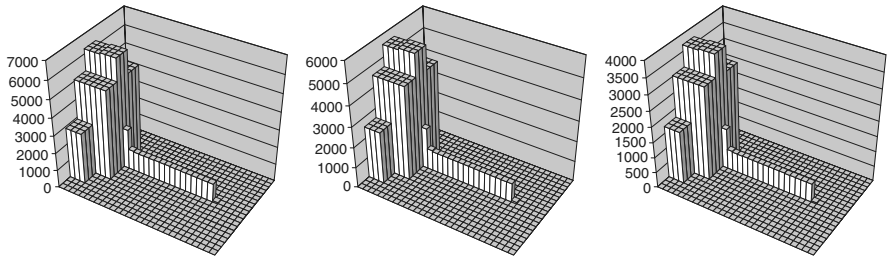

Fig. 17 Evolution over time (from *left* to *right*) of the caches of the readers in the third scenario with a maximum of 10,000 tags in the system

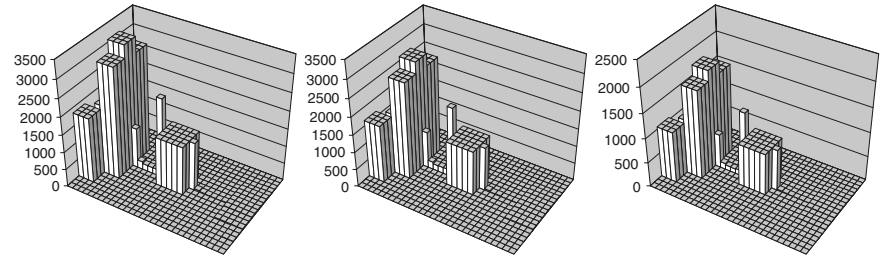

Fig. 18 Evolution over time (from *left* to *right*) of the caches of the readers in the fourth scenario with a maximum of 10,000 tags in the system

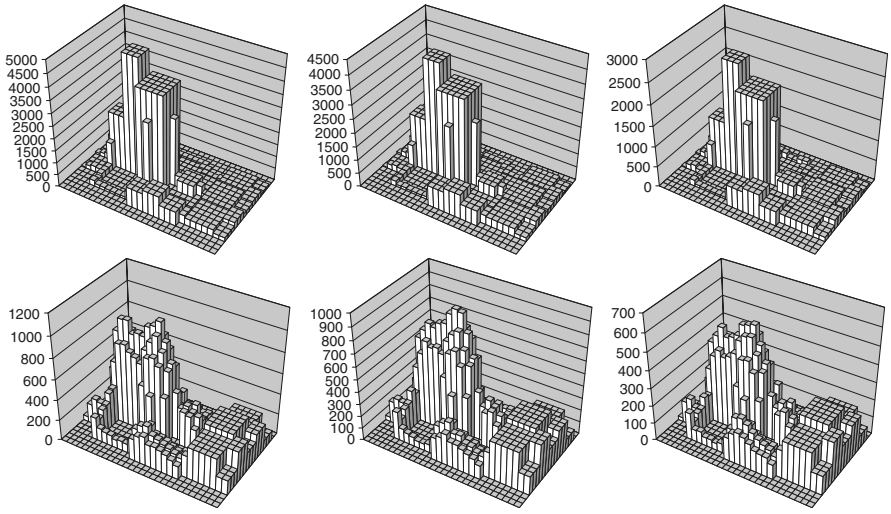

Fig. 19 Evolution over time (from *left* to *right*) of the caches of the readers in the fifth scenario with a maximum of 10,000 tags in the system. In the first row, most of the readers have cover ranges of 9 or 25 squares while in the second row most of the readers have a 1 square cover range

scenario. We have analysed two different mixtures of readers. First, we have studied the scenario covered by readers having a cover range of 9 and 25 squares (*cf.* first row of Fig. 19). Second, we have modified the readers configuration and we have mainly used 1 square cover range readers (*cf.* second row of Fig. 19). We have introduced 10,000 tags into the system to study the relation between the cover range of the readers, their cache size and the scalability.

We observe that in the first configuration the average cache size is 930.16 while in the second it is 331.80. In both cases the system scales properly (i.e. in the worst case the cache size is ten times smaller than the cache that a centralised system would need). It is clear that the use of readers having a higher cover range makes the average cache size to grow. However, we have shown that it is possible to use a mixture of reader configurations while maintaining the scalability of the system. Thus, it is possible to find the best configuration in terms of scalability (i.e. cache

size) and cost (i.e. number of readers used). In our study, we have used 65 readers in the first configuration and 137 in the second one. Depending on the scenario and on the prices of the readers, it would be easy to find the most efficient configuration in terms of scalability and cost.

6 Conclusions

In this chapter, we have tackled the problem of providing RFID systems with scalability. We have shown that scalability is a major problem that must be addressed if we want the RFID systems massive deployment to be a reality.

We have briefly analysed the most relevant techniques for providing RFID systems with security and privacy, and we have recalled a proposal for making some of these techniques scalable. We have presented our RFID simulator in detail and we have shown several experimental results that proof the usefulness of our simulator.

From the experimental results we can conclude that the proposed protocol is scalable even when the cover range of the readers increases. Moreover, we have seen that it is useful to place high cover range readers in the bottlenecks because they do not increase the average cache size significantly.

7 Future Work

The work that we are planning for the future is mainly centred in the improvement of the simulator. We want to design a new module for controlling the movement of the tags (currently it is random). We plan to allow the user to define predetermined paths which the tags will be forced to follow.

We also plan to modify the kernel in order to let it work with physical constraints associated with the used readers. Moreover, we will modify the simulator to allow the existence of overlaps amongst the readers.

Some improvements on the communication protocol are also envisaged. We believe that the number of messages exchanged by the readers could be reduced by changing the adjacency graphs used.

Disclaimer and Acknowledgements

The authors are solely responsible for the views expressed in this paper, which neither necessarily reflect the position of UNESCO nor commit that organisation. This work was partly supported by the Spanish Ministry of Education through projects TSI2007-65406-C03-01 “E-AEGIS” and CONSOLIDER CSD2007-00004“ARES”, and by the Government of Catalonia under grant 2005 SGR 00446.

The authors thank Dr. Antoni Martínez-Ballesté for his comments and suggestions.

References

1. G. Avoine and P. Oechslin. A scalable and provably secure hash based RFID protocol. In F. Stajano and R. Thomas, editors, *The Second IEEE International Workshop on Pervasive Computing and Communication Security – PerSec 2005*, pp. 110–114, IEEE, IEEE Computer Society Press, Washington, DC 2005
2. L. Batina, J. Guajardo, T. Kerins, N. Mentens, P. Tuyls, and I. Verbauwhede. An elliptic curve processor suitable for rfid-tags. *Cryptology ePrint Archive*, 2006/227, 2006. `http://eprint.iacr.org/`
3. L. Batina, J. Guajardo, T. Kerins, N. Mentens, P. Tuyls, and I. Verbauwhede. Public-key cryptography for RFID-tags. In *Printed handout of Workshop on RFID Security. RFIDSec 06*, pp. 61–76, 2006. `http://homes.esat.kuleuven.be/lbatina/Batina_RFID06_2.pdf`
4. S.J. Engberg, M.B. Harning, and C.D. Jensen. Zero-knowledge device authentication: Privacy and security enhanced RFID preserving business value and consumer convenience. In *Second Annual Conference on Privacy, Security, and Trust*, 2004
5. C. Floerkemeier, R. Schneider, and M. Langheinrich. Scanning with a purpose – supporting the fair information principles in RFID protocols. In *Proceedings of Second International Symposium on Ubiquitous Computing Systems UCS*, 2004
6. N. Good, J. Han, E. Miles, D. Molnar, D. Mulligan, L. Quilter, J. Urban, and D. Wagner. Radio frequency identifcation and privacy with informarion goods. In *Workshop on Privacy in the Electronic Society (WPES)*, 2004
7. C.C. Haley. Are you ready for RFID? *Internetnews.com*, November 2003. `http://www.internetnews.com/wireless/article.php/3109501`
8. M. Hellman. A cryptanalitic time-memory tradeoff. *IEEE Transactions on Information Theory*, IT-26: 401–406, 1980
9. D. Henrici and P. Müller. Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers. In Ravi Sandhu and Roshan Thomas, editors, *IEEE International Workshop on Pervasive Computing and Communication Security? PerSec 2004*, pp. 149–153, Orlando, Florida, USA, IEEE, IEEE Computer Society Press, Washington, DC, March 2004
10. S. Inoue and H. Yasuura. RFID privacy using user-controllable uniqueness. In *RFID Privacy Workshop,* MIT, Cambridge, MA, November 2003
11. A. Juels. Minimalist cryptography for low-cost RFID tags. In *In Fourth International Conference on Security in Communication Networks-SCN 2004*, volume 3352, pp. 149–164, Springer, Berlin, 2004. Lectures Notes in Computer Science
12. A. Juels. RFID Security and Privacy: A Research Survey. Manuscript, September 2005
13. A. Juels and R. Pappu. Squealing euros: Privacy protection in RFID-enabled banknotes. In R. Wright, editor, *Financial Cryptography'03*, volume 2742, pp. 103–121, Springer, Berlin, 2003. Lectures Notes in Computer Science
14. A. Juels and S.A. Weis. Defining strong privacy. *IACR eprint*, April 2006
15. A. Juels, R.L. Rivest, and M. Syzdlo. The blocker tag: Selective blocking of RFID tags for consumer privacy. In V. Atluri, editor, *Eighth ACM Conference on Computer and Communications Security*, pp. 103–111, 2003
16. M. Ohkubo, K. Suzuki, and S. Kinoshita. Efficient hash-chain based RFID privacy protection scheme. In *International Conference on Ubiquitous Computing – Ubicomp, Workshop Privacy: Current Status and Future Directions*, 2004
17. T. Okamoto. Probably secure and practical identification schemes and corresponding signature schemes. In E.F. Brickell, editor, *Advances in Cryptology – CRYPTO'92*, volume 740 of *LNCS*, pp. 31–53, Springer, Berlin, 1992
18. M. Rieback, B. Crispo, and A.S. Tanenbaum. RFID guardian: A battery-powered mobile device for RFID privacy management. In Colin Boyd and Juan Manuel Gonzlez Nieto, editors, *Australasian Conference on Information Security and Privacy – ACISP 2005*, volume 3574, pp. 184–194, Springer, Berlin, 2005. Lecture Notes in Computer Science

19. S.E. Sarma, S.A. Weiss, and D.W. Engels. RFID systems, security and privacy implications. Technical Report, MIT-AUTOID-WH-014, AutoID Center, MIT, 2002
20. A. Solanas, J. Domingo-Ferrer, A. Martínez-Ballesté, and V. Daza. A distributed architecture for scalable private RFID tag identification. *Computer Networks*, 51(9): 2268–2279, June 2007. http://dx.doi.org/10.1016/j.comnet.2007.01.012
21. Sybase. http://www.ianywhere.com/products/rfid_anywhere.html
22. G. Tsudik. YA-TRAP: Yet another trivial RFID authentication protocol. In *Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06)*, pp. 640–643, 2006
23. P. Tuyls and L. Batina. Rfid-tags for anti-counterfeiting. In D. Pointcheval, editor, *Topics in Cryptology-CT-RSA 2006*. Springer, Berlin, 2006. Lecture Notes in Computer Science
24. S. Weis, S. Sarma, R. Rivest, and D. Engels. Security and privacy aspects of low-cost radio frequency identification systems. In W. Stephan D. Hutter, G. Müller and M. Ullmann, editors, *International Conference on Security in Pervasive Computing – SPC 2003*, volume 2802, pp. 454–469. Springer, Berlin, 2003