RFID Privacy Using Spatially Distributed Shared Secrets

Marc Langheinrich[1] and Remo Marti[2]

[1] Inst. for Pervasive Computing
ETH Zurich, 8092 Zurich, Switzerland
langheinrich at inf.ethz.ch

[2] Ergon Informatik AG
8008 Zurich, Switzerland
remo.marti at ergon.ch

Abstract. Many of today's proposed RFID privacy schemes rely on the encryption of tag IDs with user-chosen keys. However, password management quickly becomes a bottleneck in such proposals, rendering them infeasible in situations where tagged items are repeatedly exchanged in informal (i.e., personal) situations, in particular outside industrial supply-chains or supermarket checkout lanes. An alternative to explicit access control management are RFID privacy systems that provides access to tag IDs *over time*, i.e., only after prolonged and detailed reading of an item. Such themes can minimize the risk of unwanted exposure through accidental read-outs, or offer protection during brief encounters with strangers. This paper describes a spatially distributed ID-disclosure scheme that uses a (potentially large) set of miniature RFID tags to distribute the true ID of an item across the entire product surface. We introduce the underlying mechanism of our spatially distributed RFID privacy system and report on initial performance results.

1 Introduction

Today's best protection from unwanted RFID readouts is to completely disable the tag – either by executing a *kill-command* [1] at checkout that renders the tag silent to all reader requests, or by physically clipping the tag antenna [2]. In the future, however, additional services such as warranty returns and repairs, smart laundry machines, automated inventories, or electronically augmented everyday appliances [3] may offer tangible consumer benefits for RFID-tagged items beyond the supply chain, which would force consumers to choose between these novel services and their privacy.

Short of killing tags completely, so far only password-based methods have seemed feasible for protecting RFID tags from unwanted readouts [4–6].[3] While their general principle is easy enough for implementation on a tiny RFID tag, the practical use of such schemes is often challenging. In order to facilitate the

[3] An excellent overview of RFID privacy methods can be found in [7].

exchange, sale, or return of tagged items, all involved parties must own and operate reasonably sophisticated information infrastructures that can pass and receive the individual passwords for each tagged item. In principle, NFC-enabled smartphones could easily receive such passwords as an integral part of a mobile phone based payment procedure, but in reality, it will still take many years before a majority of shoppers will own, carry, and use such phones. Equally unlikely is the fast spread of corresponding NFC-enabled point-of-sales systems, as retail-chains would need to add costly upgrades to their systems without clear benefits to their bottom line, while smaller outlets such as kiosks or newsstands would need to upgrade their entire procurement, inventory, and sales operations at costs that could easily dwarf their yearly profits.

A number of password-less alternatives for RFID privacy have since been proposed, such as Juels et al.'s *blocker tag* [8], where a specifically engineered RFID tag causes signal collisions with all regular RFID tags in its vicinity, effectively preventing their readout. While simple in use, the need for carrying a blocker tag puts the burden of privacy protection on the user, who looses this protection should she forget to carry it. Blocker tags are also subject to the same reliability concerns as ordinary tags, i.e., a suboptimal position in the reader's field might not sufficiently power the tag, thus allowing full access to all other RFID tags. In order to limit the types of deactivated tags, e.g., to only those belonging to the user, a password management scheme is again needed that allows configuring regular RFID tags to be protected by a particular blocker tag. Fishkin et al. [9] instead propose a simple but intuitive *distance-based* access control scheme, where tags reply with different levels of detail depending on their distance to the reader. Apart from the increased costs for the required on-tag circuitry to reliably detect signal strength, distance-based authentication might not always yield the desired functionality, e.g., when passing narrow passageways or small store entrances.

In an earlier paper [10], we have proposed a third alternative, called a *Shamir Tag*, which neither require costly password management nor error-prone distance measurements. Using the cryptographic principle of *secret shares* [11], Shamir Tags yield virtually no information to casual "hit-and-run" attackers, but only reveal their true ID after continuous and undisturbed reading from up-close – something that can hardly go unnoticed by an item's owner. At the same time, however, the system allows tag owners to use *caching* for speeding-up this process, effectively preserving instant item identification in home-automation or supply-chain applications.

In order to prevent secret long-range scanning with powerful antennas, Shamir Tags' antennas will need to be constructed with limited read-out ranges, potentially yielding only a few centimeters of distance for systems operating within the allowed power levels. This in turn might complicate the readout process also for tag owners, as tagged items need to be positioned more carefully with respect to the antenna. This paper presents a *multi-tag* extension to Shamir Tags, allowing the use of dozens, if not hundreds of miniature tags on the same product, thus alleviating positioning problems without the need for increased read ranges.

Our approach is based on the idea of *super-distributed RFID tag-infrastructures* (SDRI) [12], where tiny RFID chips are brought out in large numbers, e.g., mixed into wall paint, woven into carpets or clothing, or sprinkled into an item's plastic casing. Thus, instead of having a single RFID tag per item, we envision items that feature several hundreds of tags, with the item's ID being *spread out* across all tags. Given appropriate communication protocols and antenna sizes, reading that many tags at once will be infeasible, instead requiring readers to scan small areas sequentially. While clearly not yet a reality, we believe that current trends in RFID miniaturization, such as Hitachi's μ-chip,[4] offer ample potential for actually deploying such simple but reliable RFID privacy systems in the future.

The remainder of this paper is structured as follows. Section 2 will briefly describe our previously proposed Shamir Tags and their underlying principles, Shamir's *secret sharing* scheme and *bit-throttling*, as well as outline a distributed, multi-tag variant. Section 3 then presents two extensions that we developed for using distributed Shamir Tags concurrently, i.e., in a multi-item scenario. Section 4 will briefly outline the prototype system we built for evaluating our approach, before we report on the results of initial experiments in section 5.

2 Shamir Tags

Shamir Tags use two principles to protect the true ID of an item (e.g., its EPC-code) [10]. Firstly, data readout is performed in two stages using a *bit-by-bit* strategy. Initially, a Shamir Tag discloses a small subset (e.g., 5%) of bits to a reader, which allows owners to quickly identify the entire bit-string from a small list (cache) of personal items. This is then followed by a steady "trickle" of bits that reveals the entire ID to the reader only after prolonged reading, e.g., several minutes. This allows anybody to eventually identify an item, yet forces them to stay close enough to the tag during the entire time. This process is called *bit-throttling*, and it makes tag-tracking difficult.

However, since industrial code-schemes are often heavily structured, even releasing only a few bits might already disclose sensitive data. E.g., an EPC-header featuring the combination `10` at the third and fourth position uniquely identifies items tagged by the U.S. Department of Defense [13]. To prevent such data disclosure, Shamir Tags are additionally *encoded* using *shared secrets*. The process of creating a shared secret basically re-encodes the tag's true ID into n seemingly unrelated numbers. Only by combining all n numbers, the original ID can be (trivially) reconstructed. Section 2.1 will give some more background information on this process – for now it suffices to know that this encoding step basically protects our Shamir Tag from inadvertently disclosing meaningful bits. Only after all bits have been read (which, due to bit-throttling, may take up to several minutes) can they be combined into the true ID.[5]

[4] Hitachi's current generation μ-chip has a size of less that 0.2 mm^2, its next generation will have only about 0.02 mm^2. Also see `www.hitachi.co.jp/Prod/mu-chip`.

[5] Note that if x bits are missing, rogue readers can of course try out all possible 2^x combinations to compute 2^x potential true IDs, and then use knowledge about

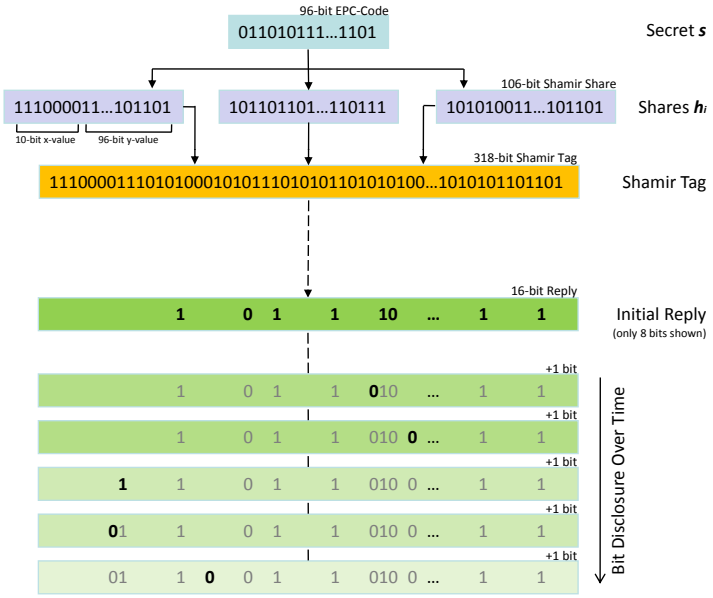


Fig. 1. *Principal Construction of a Shamir Tag* (from [10]). Based on the tag's "true" ID, e.g., its EPC-code, multiple Shamir shares are concatenated to form the tag's new ID, which is then stored on the tag. Upon reader inquiry, an initial set of random bits is released, with subsequent throttled single-bit releases.

Figure 1 shows the principal construction of a Shamir Tag from a 96-bit EPC. In our previous work [10], we have shown that Shamir Tags provide an effective and cheap protection from unwanted and inadvertent tag readouts. Item owners can use simple caching strategies to ensure instantaneous identification of their own tags, while foreign readers will need to have continuous access to the tag for prolonged amounts of time, in order to read a sufficiently large percentage of bits from the tag that allows reconstructing the Shamir-encoded true ID. However, a critical factor of this protection is the effective read range of such tags – if the read range is too large, attackers can read out tags from several meters away whenever their owners are not moving fast enough, e.g., in public transport, or while waiting in line. Reducing the read range by limiting tag antenna sizes helps to prevent such attacks, yet at the same time complicates tag readout for legitimate owners, as they will also need to position their antennas very close to the tag. In industrial settings, or when the exact location of an embedded tag is not known, this might significantly hamper legitimate tag use.

Our solution to this is – as outlined in the introduction – straightforward: instead of using a single Shamir Tag with a reasonable antenna range that sim-

valid EPC values (e.g., allowed manufacturer IDs, or known product IDs) to discard invalid IDs.

plifies tag detection at the expense of long-range scanning protection, we use dozens, if not hundreds of miniature Shamir Tags, woven into the garment of clothing, or mixed into the plastic casing of products, that have a much shorter antenna range but which distribute the item's (protected) ID more or less evenly across the entire product surface. However, this approach offers new challenges for ID reconstruction, which are outlined in section 3 below. But first, we will briefly give some background on the construction of shared secrets using Shamir's scheme in the following section.

2.1 Shamir's Secret Sharing Scheme

In a secret sharing scheme, each participant receives a *share* that is a part of a secret. The secret can only be recovered if enough participants cooperate in recombining their shares. Schemes that allow a reconstruction of the secret with only t out of n participants involved are called *(t,n)-threshold schemes*. They fulfill the following two properties: Firstly, no subset of participants smaller than a threshold t can gain information on the secret s, even when cooperating with each other. Secondly, any subset equal to or larger than a threshold t can reconstruct the secret s at any time.

One of the most famous (t,n)-threshold schemes was introduced by Shamir in 1979 [11]. It is based on polynomials, and in particular on the observation that a polynomial of degree $t-1$ is defined by t coordinate-pairs (x_i, y_i). To encode a secret s for n participants with a threshold t, one chooses a random polynomial of degree $t-1$ that crosses the y-axis at s. The n participants are each given exactly one point on the polynomial's curve, thus allowing any t members to compute the exact polynomial and thus the y-intercept s.

The reconstruction of the secret is essentially a polynomial interpolation based on the *Lagrange* formula. Since only the y-intercept is of interest, it can be simplified to the following formula (with k being the number of tags read):[6]

$$s = q(0) = \sum_{i=1}^{k} y_i \prod_{1 \le j \le k, i \ne j} \frac{x_j}{x_j - x_i} \tag{1}$$

In practice, computing the secret s given large numbers of shares (e.g., thousands) quickly becomes infeasible. Calculations are therefore carried out in a finite field modulo p (written as $\mathbb{Z}_p$)[7], with p being a large prime number. Not only does this reduce the size of exponents, but it also removes the need for floating point operations (thus limiting numerical errors).

A comprehensive discussion of this topic is beyond the scope of this paper, but an excellent introduction, as well as efficient algorithms for solving (1) in $\mathbb{Z}_p$, can be found in [14]. Operating a secret sharing scheme within $\mathbb{Z}_p$ not only makes reconstruction of the secret s (e.g., its Electronic Product Code/EPC) feasible, but also helps with the practical problem of resolving multiple secrets concurrently. This will be described in section 3 below.

[6] Obviously, computing s with $k < t$ shares is not possible.

[7] $\mathbb{Z}_p$ designates the set $\{0, 1, ..., p-1\}$.

2.2 A Spatially Distributed Shamir Tag

A straightforward implementation of a distributed Shamir Tag would simply put the individual shares not just on a single tag, but distribute them among *multiple* tags on (or in) an item. As Shamir's scheme allows the reconstruction of the secret irrespective of the order of the shares, no special order would need to be observed when reading shares off the individual tags. Bit-throttling could also still be used, as each tag would choose a random temporary ID during readout, allowing a reader to group bits from the same share properly together. In order to make use of *caching* [10], however, bits would need to be continously numbered across all tags, in order to have a defined order. Note that this would *not* decrease the level of protection compared to a single Shamir Tag, as this simply orders the distributed bits just as in the non-distributed (i.e., single-tag) version – this would simply increase per-tag storage requirements, as each distributed share would need to also store its original position within the Shamir Tag.

By properly adjusting the threshold parameter t, defective or detuned tags could be tolerated. This also adds flexibility to the readout process, as only part of an item's surface would need to be scanned.[8] Just like in the single-tag case, a reader would gradually assemble the set of tags and their IDs in an item and repeatedly compute the secret s until a stable y-intercept had been found. Obviously, the overall disclosure-delay of a single tag could be significantly shortened, as the spatial distribution of the shares combined with the shortened read range of individual tags introduces an additional delay during readout.

3 Distributed Multi-Item Identification

The approach described in section 2.2 above works well as long as only a single item/ID at a time needs to be reconstructed. However, once multiple items are within the reading range of the antenna, interpolation points from two or more polynomials would get mixed together that would never converge on a stable s value (nor yield multiple values for the different items). Since the Shamir scheme has no means of differentiating points from different polynomials, we will need to extend it if we want it to support decoding two or more secrets concurrently.

A naïve idea to discriminate between different set of interpolation points would be to use a common prefix for all tags of a single item, thus allowing a reader to compute multiple s-values for different items concurrently. Obviously, such a prefix would constitute just another fixed, trackable pseudonym, rendering the benefits of the entire sharing scheme void. Instead, we will need a method similar to our initial approach, i.e., a discrimination system that works well once a certain threshold of points have been assembled, but which does not work if only few interpolation points have been read.

[8] The ratio between t and n could be adjusted individually for different products, depending on the envisioned privacy degree: a threshold t close to n requires many tags to be read, a small t allows the reconstruction of the secret s already with a small subset of tags.

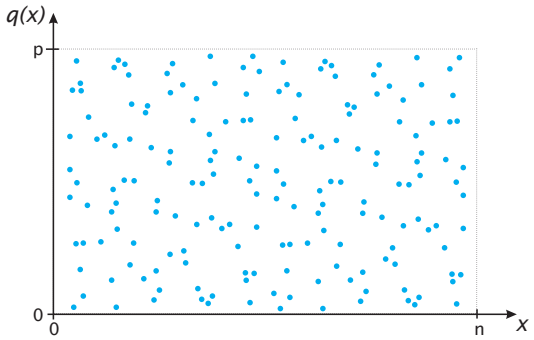


Fig. 2. *Distribution of Interpolation Points in $\mathbb{Z}_p$.* The sampling points on the polynomial $q(x)$ *mod* p appear evenly distributed in the Shamir Space $n \times p$. This allows us to choose a subset n' of points that fulfill certain geometric requirements, thus facilitating item discrimination. Note that the drawing is not to scale, as typically $p \gg n$.

The following sections describe two such discrimination methods that we have developed, both based on the geometric distribution of interpolation points: *cluster-based discrimination* and *line-based discrimination*. They both make use of the fact that the n interpolation points on the polynomial $q(x)$ *mod* p (i.e., within $\mathbb{Z}_p$) are spread in a seemingly uniform way in the "Shamir Space" $n \times p$ (see figure 2). The basic idea is to oversample $q(x)$ with much more points than needed for item integration. We then have the choice of carefully selecting n' out of the n generated points, based on the specific geometric requirements of our discrimination method, and use only those n' points as our tags.[9] During item detection, we can then use the discrimination method to properly distinguish tags from different items.

3.1 Cluster-Based Item Discrimination

For the *clustering method*, we selects the n' points in the shape of several clusters, all of them similar in size and laid out on a regular grid. This allows our cluster-based discrimination algorithm to determine which tag-IDs belong to the same item, according to cluster size and cluster position. Figure 3 illustrates the selection of IDs for one item during tag generation. In addition to the previously discussed parameters (i.e., the total number n' of tags on the item, the encoded secret ID s, and the threshold of tags t that need to be read), each item gets assigned a random number c of clusters and a corresponding cluster size[10] (len_x and len_y). Given a system-wide, fixed cluster grid size (d_x and d_y) and

[9] Note that $n \gg n' > t$ holds, i.e., the secret can still be computed with only t tags.

[10] The cluster dimensions have to be chosen in such a way that the expected sum of all points contained therein matches or slightly exceeds n'. Under the assumption of regular point distribution, cluster length and width can be calculated as $len_x = \sqrt{\frac{n'}{n \cdot c}} \cdot n$ and $len_y = \sqrt{\frac{n'}{n \cdot c}} \cdot p$. Also note that p is usually much larger than n, resulting

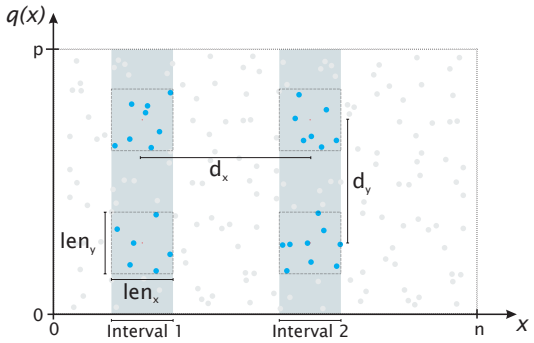


Fig. 3. *Cluster-Based Selection of n'.* Given a choice of individual item parameters d_x and d_y, $c = 4$ clusters of approximately the same size (in an area of $len_x \times len_y$) are randomly selected from a regular grid.

the Shamir Space dimensions n and p, the tag-ID generation algorithm simply chooses a random origin and then proceeds to place the c clusters randomly on a rectangular grid. Note that instead of sampling the polynomial for all n x_i, it suffices to compute the y_i within the intervals where the actual clusters lie, thus greatly speeding up share generation. These intervals are also shown in figure 3.

Once we have generated the tag-IDs according to the above steps, *item detection* can then proceed in three phases: cluster detection, item discrimination, and secret reconstruction. In the first phase, all found tag-IDs are analyzed by means of a clustering algorithm in order to identify cluster centers. We have found the *subtractive clustering* algorithm [15] to be very efficient for our purposes. Note that an appropriate distance measure must be chosen in order to account for the stretched space with its elongated clusters. Once the clusters have been returned by the algorithm, phase two groups clusters with similar size (i.e., number of associated points) and position (i.e., with the cluster center near a node on the same grid) together. Since tags for different items are generated independently, cluster collisions can occur if two items in the reading range have tags residing in the same area. Such a scenario is depicted in figure 4. Collisions result in larger clusters and are discarded by the detection algorithm as they do not match any other cluster either with regard to size, position, or both. However, in most cases this will still leave enough points for reconstruction, as t out of n' tags already suffice. Phase three finally reconstructs the secrets s_i separately for all identified items I_i by means of (1) within $\mathbb{Z}_p$.

3.2 Line-Based Discrimination

The *line-based method* locates the subset of n' tags per item along lines of different origin and slope within the oversampled Shamir Space. In order to fa-

in a massively stretched Shamir Space. In account with this stretch, len_x and len_y are chosen such that the ratio $\frac{len_x}{len_y}$ equals $\frac{n}{p}$.

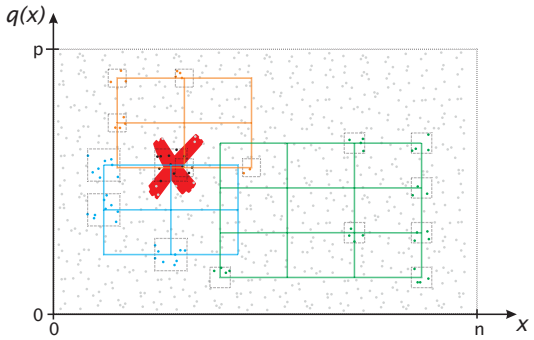


Fig. 4. *Cluster-Based Item Detection in the Shamir Space.* The example shows clusters from three different items. A collision between two clusters occurs that can be identified due to the unusual size and position of the resulting collision-cluster.

cilitate detection, we restrict lines to a number of four predefined slopes $m \in \{0, 1, -1, \pm\infty\}$[11]. While the slope m of an item's line is selected randomly, the line's respective y-intercept b depends on m, as shown in table 1. A third parameter, the line width d (i.e., the maximal allowed distance of a point from the line), is used to regulate the number of points available along a line, as we eventually need to select n' tags per item, irrespective of line slope and origin.[12]

In contrast to the cluster-based detection, the line width parameters d^i (i.e., one for each possible value of m) must be chosen and fixed in advance for all items in the system, irrespective of an item's number of shares n'. This is because during detection, we otherwise have no way of bounding our search along these lines. Given the set of four slopes and y-intercepts shown in table 1, the following widths guarantee that all types of lines select at least n'_{max} tags[13] in the Shamir Space:[14]

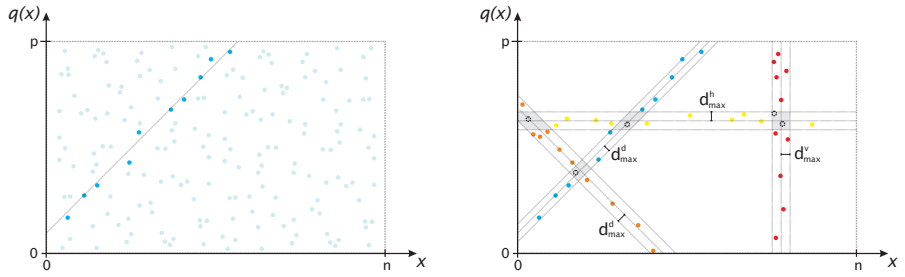
$$d^v_{max} = tolerance, \quad d^h_{max} = \frac{p}{n} \cdot tolerance, \quad d^d_{max} \geq \frac{p \cdot tolerance}{\sqrt{\left(\frac{p}{2}\right)^2 + \left(\frac{n}{2}\right)^2}},$$

[11] Since $p \gg n$ holds, we actually use $m \in \left\{0, \frac{p}{n}, -\frac{p}{n}, \pm\infty\right\}$, as the slopes ± 1 and 0 can hardly be distinguished.

[12] This is necessary as different slopes and different origins result in widely different numbers of available points. A horizontal line ($m = 0$) has a length of only n, while a vertical line ($m = \pm\infty$) extends over p units (recall that $p \gg n$). Their respective widths d^h_{max} and d^v_{max} must compensate for their difference in length in order for both to select n' points. Diagonal lines ($m = \pm\frac{p}{n}$) differ greatly based on their origin: The more centered they are in the Shamir Space, the more points are selected. The closer they are to the corners, the fewer points are available. However, we cannot always choose lines in the center, as this would cause different lines to collide with high probability, rendering item detection almost impossible.

[13] If a smaller amount than n'_{max} is desired, points can be discarded randomly, effectively *thinning* the line.

[14] In equation 2, n'_{max} designates the largest n' of all items ever to be generated.



(a) *Selection.* n' tags are selected randomly along lines with slope m, width d, and y-intercept b.

(b) *Item Detection.* The example shows lines from four different items. Tags in the intersecting areas are discarded.

Fig. 5. *Line-Based Item Discrimination.* Lines representing items cross the Shamir Space with predefined slopes.

$$\text{with} \quad tolerance = \frac{n'_{max}}{2} \tag{2}$$

Knowing the system-wide *tolerance* parameter, item discrimination is fairly straightforward. After acquiring the point cloud of all tags in the reading range, the detection algorithm selects a random point and searches for close-by values in all possible directions (i.e., horizontal, vertical and diagonal). Should the number of detected points in the four directions not differ significantly, the starting point most likely lies in an intersection of two or more lines, which prompts the algorithm to choose another random point and start over. As the starting point might reside near the edge of a line, the four directions are explored with a distance of $2d_{max}$[15] (see figure 5(b)). An pruning step then discard points that have wrongly been assigned due to this increased search range. For diagonal lines, this is done by trying to estimate the real line using a *linear least squares* fit and rejecting points in a distance larger than d^d_{max}. For horizontal and vertical lines, simply averaging over the points' (x,y)-coordinates results in the estimated line. Points with a distance larger than d^h_{max} ord^v_{max}, respectively, are discarded.

All remaining points after the pruning step are considered valid points of an item, and are subsequently removed from the point cloud before the algorithm is restarted for detecting another item among the remaining points. Once all points have been assigned to items this way, the algorithm finally checks for intersections. Points in the area of an intersection will be removed from the corresponding items' sets, as they might have been wrongly assigned to an item. A final phase reconstructs the s_i by means of (1) within $\mathbb{Z}_p$ for all items.

[15] That is, $2d^h_{max}$ for horizontal, $2d^v_{max}$ for vertical and $2d^d_{max}$ for diagonal lines.

Table 1. *Parameter Choices for Line-Based Detection.* Depending on a line's slope m, different y-intercepts b are chosen.

Slope m	0	$\frac{p}{n}$	$-\frac{p}{n}$	$\pm\infty$
y-intercept b	$[d^h_{max}, p-1-d^h_{max}]$	$[-\frac{1}{2}p, \frac{1}{2}p]$	$[\frac{1}{2}p, \frac{3}{2}p]$	$[d^v_{max}, n-d^v_{max}]$

4 Prototype System

In order to evaluate our proposed time-delayed identification method, we created a Java application that could both simulate tag generation and item detection for arbitrarily large item and tag populations, as well as drive an actual hardware reader (using Hitachi's μ-Chips)[16] to read out SDRI-tagged items. Figure 6 shows the main program interface, as well as an actual set of SDRI-tagged items (baby clothing) that can be read out using a regular reader.[17]

The SDRI-Privacy Demonstrator allows manual and automated generation of tag-sets, i.e., the encoding of arbitrary IDs s_i (e.g., an EPC) onto arbitrary numbers of tags n_i with a chosen threshold t_i. To simplify the operation of our initial prototype, we have not implemented the bit-throttling feature of the Shamir tags – upon readout, each tag reveals one of m_i Shamir shares stored on the tag. The generated shares can be assigned to simulated tags (for simulated items), or linked to real RFID-tags (that are affixed to real items) in the form of a translation table, which translates the fixed ID of a read-only RFID-tag into one of the m_i Shamir shares of the Shamir polynomial.[18]

A built-in simulator can be used to automate the process of repeatedly reading out RFID-tags from the generated item sets, keeping track of successful item identification under various conditions. The results presented in section 5 below are based on such simulations. For demonstration purposes, a small number of tags can also be read directly using a conventional RFID reader. Due to their small size, short range, and restricted anti-collision capabilities, a few dozen Hitachi μ-Chips were incorporated into a small set of baby clothes (see figure 6(b)[19]), demonstrating the envisioned interaction concept: only by sweeping the reader back and forth over the items, the individual IDs of each clothing can be reassembled.[20]

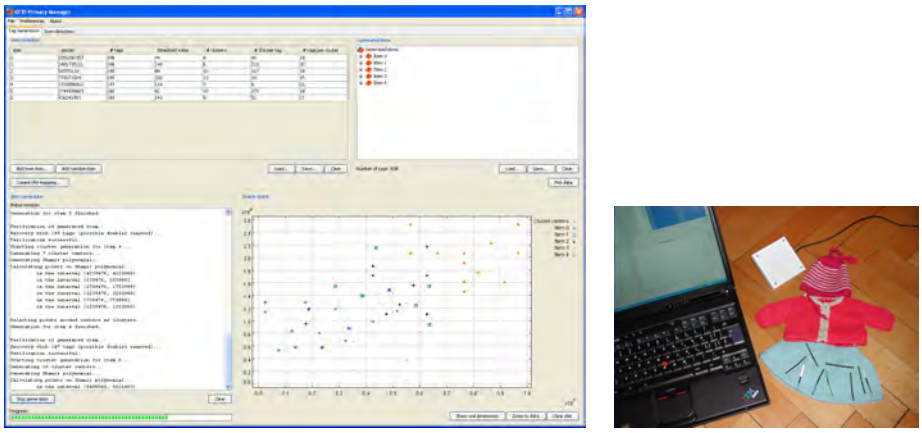
[16] See `www.hitachi.co.jp/Prod/mu-chip`

[17] While both the *clustering* and the *line* method were prototypically implemented in `MATLAB`, the demonstrator currently supports only the clustering method.

[18] Ideally, the shares on a tag (representing interpolation points on its item's polynomial) are set during the manufacturing process, e.g., using write-once tags. However, as the RFID tags we were using were not writable, their fixed IDs are mapped to coordinates in Shamir space by means of a translation table that is maintained in the demonstrator software.

[19] The skirt is pictured inside out, showing the affixed μ-chip tags.

[20] Note that the short antenna sizes on the RFID chips and the lack of an anti-collision protocol are central to our approach, as they prevent malicious readers from quickly scanning all available tags on a person, potentially from a large distance.



(a) Screenshot of Main Program Interface (b) Setup in Hardware Mode

Fig. 6. *The SDRI-Privacy Demonstrator.* A prototypical implementation demonstrates the feasibility of the SDRI-privacy approach and allows performance measurements.

5 Analysis

This section will evaluate the performance of our proposed spatially distributed Shamir Tags in terms of *detection rates* and *traceability*. Ideally, we want our tagged items to be difficult to trace, yet reliably detectable for authorized readers.[21]

Detection might be hampered by the bit-throttling of each individual tag, as well as due to our multi-item discrimination algorithms (i.e., the cluster-based and line-based methods described in section 3). While we have not yet implemented bit-throttling in our prototype, the general performance aspects of our previously proposed *single-tag* solution [10] should still hold. Table 2 (reprinted from [10]) shows the discriminatory power of a certain number of bits in a cached population. Using this, we can conclude that a set of 10 distributed tags each releasing only 1-2 bits upon initial readout (thus yielding about 15 bits total) would already allow the identification of an item from a list of some 30 000 items (see row 15 and interpolate the values between column 10 000 and 100 000). This of course requires unique bit-positions among all tags on an item, in order to allow for this kind of lookup table to work. Note, however, that this does not imply that tags would be *traceable* using these 15 bits: it is only if a tag owner knows the entire set of Shamir shares that this lookup works – simply knowing 15 bits of an item obviously does not allow identification without knowing *all* bits.

This leaves us to evaluate our multi-item discrimination algorithms, which might not be able to properly separate shares/points from multiple items. We

[21] As we do not use passwords, "authorized" in our case means a reader that is able to read a majority of the tags (which should take considerable effort).

Table 2. Number of Items Identified by Bit-Strings of Different Lengths (from [10])

Bits ↓ Items →	100	1 000	10 000	100 000	1 000 000	10 000 000	100 000 000	1 000 000 000	10 000 000 000
1	50	500	5 000	50 000	500 000	5 000 000	50 000 000	500 000 000	5 000 000 000
2	25	250	2 500	25 000	250 000	2 500 000	25 000 000	250 000 000	2 500 000 000
3	13	125	1 250	12 500	125 000	1 250 000	12 500 000	125 000 000	1 250 000 000
4	6	63	625	6 250	62 500	625 000	6 250 000	62 500 000	625 000 000
5	3	31	313	3 125	31 250	312 500	3 125 000	31 250 000	312 500 000
6	2	16	156	1 563	15 625	156 250	1 562 500	15 625 000	156 250 000
7	**0.78**	8	78	781	7 813	78 125	781 250	7 812 500	78 125 000
8	0.39	4	39	391	3 906	39 063	390 625	3 906 250	39 062 500
9	0.20	2	20	195	1 953	19 531	195 313	1 953 125	19 531 250
10	0.10	**0.98**	10	98	977	9 766	97 656	976 563	9 765 625
11	0.05	0.49	5	49	488	4 883	48 828	488 281	4 882 813
12	0.02	0.24	2	24	244	2 441	24 414	244 141	2 441 406
13	0.01	0.12	**1**	12	122	1 221	12 207	122 070	1 220 703
14	0.01	0.06	0.61	6	61	610	6 104	61 035	610 352
15	0.00	0.03	0.31	3	31	305	3 052	30 518	305 176
16	0.00	0.02	0.15	2	15	153	1 526	15 259	152 588
17	0.00	0.01	0.08	**0.76**	8	76	763	7 629	76 294
18	0.00	0.00	0.04	0.38	4	38	381	3 815	38 147
19	0.00	0.00	0.02	0.19	2	19	191	1 907	19 073
20	0.00	0.00	0.01	0.10	**0.95**	10	95	954	9 537

also need to analyze how the discrimination algorithms affect traceability. This is done in the following two sections.

5.1 Detection Rates

Since the math behind Shamir's secret sharing *guarantees* that we can successfully identify a single item once t or more pieces have been read (i.e., more than an item's threshold), we do not need to evaluate the chances for identifying a single-item, once t or more shares have been read. With multiple items, however, it is up to our discrimination algorithms presented in section 3 to properly group the individual tags into separate items. If only a single tag is accidentally assigned to the wrong item, detection will fail.[22] How often does this, on average, happen? We used our Java simulator and ran 100 iterations of the following experiment:

- Generate between 1 and i items (i being 20 or 10), each having between $0.5n'$ and n' tags (n' being 800 or 600) with a random threshold t between $0.4n'$ and $0.8n'$.
- From all generated tags, read a random fraction of f tags ($f = 0.8, 0.9, 1.0$).
- Identify items and record percentage of items identified.[23]

Table 3 shows the detection rates of both the *cluster-based* and the *line-based* method. The tests for the cluster-based method were performed with our

[22] Note that it is easy to verify whether we have assembled the right item ID, as removing a single tag from the set should not change the retrieved secret value s (unless we have collected exactly t or even fewer tags).

[23] Note that we do not have false positives, i.e., we will never wrongly identify a non-existent item (see previous footnote).

Table 3. *Detection Rates.* Simulation results for the *clustering* and *line* methods with three different setups. While the cluster-based method achieved high item discrimination rates, the line-based method is limited by a much smaller Shamir Space.

Percentage of Tags Read	Detection Rate Clustering	Detection Rate Lines	System Setup
100%	99.36%	95.12%	up to 20 items, each $\leq$ 800 tags
90%	97.38%	93.81%	
80%	94.67%	91.10%	
100%	99.66%	95.32%	up to 20 items, each $\leq$ 600 tags
90%	98.06%	94.91%	
80%	94.60%	94.15%	
100%	99.89%	98.33%	up to 10 items, each $\leq$ 800 tags
90%	99.11%	97.90%	
80%	96.20%	97.57%	

simulator, using $n = 10\,000\,000$ and $p = 3\,037\,000\,493$. The line-based method, in contrast, uses a much smaller value of $n = 120\,000$ (but the same value for p), as otherwise the sampling of horizontal or diagonal lines across the entire Shamir Space proved to be too costly. This, in turn, has a direct effect on the detection rates, as the much smaller Shamir Space in our line-based simulation results in more overlaps, where conflicting tags must be removed (and thus cannot help with item identification).

The three different test cases shown in table 3 demonstrate that detection rates are better if fewer items are within reading range. Lowering the number of tags per item only yields marginal improvements, however. This is because fewer tags only "thin out" our clusters/lines, while fewer items translate directly into fewer clusters/lines, thus reducing the number of collisions (this effect, however, is much more significant when using the line-based method, see table 3).

5.2 Traceability

One might argue that due to our clustering methods, repeated readouts might show re-identifiable cluster patterns. Figure 7 shows a set of four readouts from a person carrying 15 items, each containing up to 800 tags (totaling 9032 tags). Each read detects a random subset of 0.5% of all tags. Due to the setup of the cluster-based method, a random subset of tags is typically scattered widely across the whole Shamir space. When reading only a small subset of all integrated tags, clusters are hardly visible. This changes, however, if the system's Shamir Space (i.e., $n \times p$) is very large. Under such conditions, tags of a single cluster are grouped relatively close together, which in turn makes cluster differentiation across multiple readout simple. This is an inherent trade-off between good detection rates and the prevention of traceability in this method.

A visual inspection of the line-based method exhibits similar results. With very small subsets, the alignment of the tags along lines is no longer visible and they seem to be randomly spread across the Shamir Space. With large values of

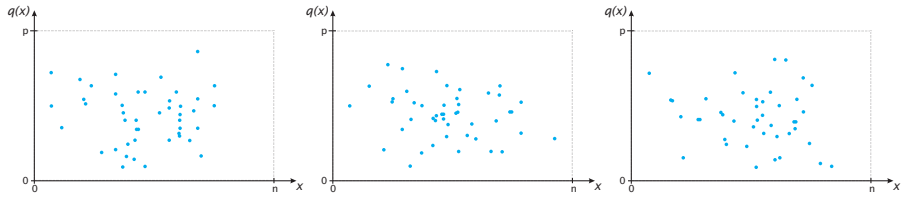


Fig. 7. *Tracking Items Without Overlapping IDs.* With no direct ID overlap, tracking people requires identifying patterns in the clustering algorithm. The above plots show three subsequent readouts of ≈45 tags each, based on 15 items with ≈800 tags.

n, however, the increased space separates the lines more clearly, resulting in a visibility of the individual items even in small tag subsets. Surprisingly though, the line-based method seems to outperform the cluster-based method in terms of traceability protection. Since a large n causes clusters to be very small, one single tag ID can easily give the position of a whole cluster away. This is different with the line-based method, as tags can, in principle, appear anywhere within the Shamir Space.

6 Conclusions

In this paper, we have extended our previously proposed alternative access control method [10] to allow for a *spatial distribution* of tags. This allows us to further limit read ranges of individual tags without making finding the tag (for readout) impossible. Instead, the surface of an item could be covered with dozens or even hundreds of tags featuring very short read ranges, thus making secret long-range readouts practically impossible.

We have extended the existing scheme based on shared secrets with support for concurrently resolving *multiple* secrets. This is achieved by means of geometric discrimination functions, two of which have been proposed in this paper: a cluster-based, and a line-based method. Simulations have demonstrated the ability of these methods to properly distinguish multiple items, while at the same time offering reasonable protection against unwanted tracking.

We envision that the continuing trend in miniaturization will one day render RFID chips the very first "smart dust" that can be cheaply woven into garments, integrated into plastic casings, or mixed into wall paint. Write-once tags might then be easily initialized during production using our methods described above and provide an implicit privacy protection without preventing any of the envisioned future uses of RFID-enabled items.

7 Acknowledgements

This work has been partially funded by Hitachi Systems Development Laboratories (SDL), Japan, who also provided the μ-chip RFID equipment.

References

1. EPCglobal: Class-1 generation-2 UHF RFID protocol for communications at 860 MHz–960 MHz, version 1.0.9. EPC radio-frequency identity protocols (2005) See `www.epcglobalinc.org/standards/Class_1_Generation_2_UHF_Air_Interface_Protocol_Standard_Version_1.0.9.pdf`.
2. Karjoth, G., Moskowitz, P.A.: Disabling RFID tags with visible confirmation: clipped tags are silenced. In Atluri, V., De Capitani di Vimercati, S., Dingledine, R., eds.: Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society (WPES 2005), Alexandria, VA, USA, ACM Press (2005) 27–30
3. Roduner, C., Langheinrich, M., Floerkemeier, C., Schwarzentrub, B.: Operating appliances with mobile phones – strengths and limits of a universal interaction device. In: Proceedings of Pervasive 2007, Toronto, Canada, May 13-16, 2007. LNCS, Berlin Heidelberg New York, Springer (2007)
4. Weis, S.A., Sarma, S.E., Rivest, R.L., Engels, D.W.: Security and privacy aspects of low-cost radio frequency identification systems. In Hutter, D., Müller, G., Stephan, W., Ullmann, M., eds.: Security in Pervasive Computing – First International Conference, Boppard, Germany, March 12–14, 2003, Revised Papers. Volume 2802 of LNCS, Berlin Heidelberg New York, Springer (2004) 201–212
5. Ohkubo, M., Suzuki, K., Kinoshita, S.: Cryptographic approach to "privacy-friendly" tags. In Garfinkel, S., Rosenberg, B., eds.: RFID: Applications, Security, and Privacy. Addison-Wesley (2005)
6. Henrici, D., Müller, P.: Tackling security and privacy issues in radio frequency identification devices. In Ferscha, A., Mattern, F., eds.: Pervasive Computing – Second International Conference, PERVASIVE 2004, Vienna Austria, April 21–23, 2004, Proceedings. Volume 3001 of LNCS, Berlin Heidelberg New York, Springer (2004) 219–224
7. Juels, A.: RFID security and privacy: A research survey. IEEE Journal on Selected Areas in Communications **24**(2) (2006) 381–394
8. Juels, A., Rivest, R.L., Szydlo, M.: The blocker tag: Selective blocking of RFID tags for consumer privacy. In Jajodia, S., Atluri, V., Jaeger, T., eds.: Proceedings of the 10th ACM Conference on Computer and Communication Security, Washington, D.C., USA, ACM Press (2003) 103–111
9. Fishkin, K., Roy, S., Jiang, B.: Some methods for privacy in RFID communication. In Castelluccia, C., Hartenstein, H., Paar, C., Westhoff, D., eds.: Security in Adhoc and Sensor Networks – First European Workshop, ESAS 2004, Heidelberg, Germany, August 6, 2004, Revised Selected Papers. Volume 3313 of LNCS, Berlin Heidelberg New York, Springer (2005) 42–53
10. Langheinrich, M., Marti, R.: Practical minimalist cryptography for RFID privacy. Submitted for publication (2007) Available online at `www.vs.inf.ethz.ch/publ/papers/shamirtags07.pdf`.
11. Shamir, A.: How to share a secret. Comm. of the ACM **22**(11) (1979) 612–613
12. Bohn, J., Mattern, F.: Super-distributed RFID tag infrastructures. In: Ambient Intelligence – Second European Symposium, EUSAI 2004. Volume 3295 of LNCS, Springer (2004) 1–12
13. EPCglobal: EPC tag data specification 1.1. EPCglobal Standard (2003)
14. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of applied cryptography. CRC Press, Boca Raton, Florida (1996)
15. Chiu, S.L.: Fuzzy model identification based on cluster estimation. Journal of Intelligent and Fuzzy Systems **2**(3) (1994) 267–278