PPS: Privacy-Preserving Statistics using RFID Tags

Erik-Oliver Blass   Kaoutar Elkhiyaoui   Refik Molva
EURECOM
2229 Route des Crêtes, BP 193
06560 Sophia Antipolis, France
{blass|elkhiyao|molva}@eurecom.fr

ABSTRACT

As RFID applications are entering our daily life, many new security and privacy challenges arise. However, current research in RFID security focuses mainly on simple authentication and privacy-preserving identification. In this paper, we discuss the possibility of widening the scope of RFID security and privacy by introducing a new application scenario. The suggested application consists of computing statistics on private properties of individuals stored in RFID tags. The main requirement is to compute global statistics while preserving the privacy of individual readings. *PPS* assures the privacy of properties stored in each tag through the combination of homomorphic encryption and aggregation at the readers. Re-encryption is used to prevent tracking of users. The readers scan tags and forward the aggregate of their encrypted readings to the back-end server. The back-end server then decrypts the aggregates it receives and updates the global statistics accordingly. PPS is provably privacy-preserving. Moreover, tags can be very simple since they are not required to perform any kind of computation, but only to store data.

1. INTRODUCTION

In Radio Frequency IDentification (RFID), tags are transponders that reply to reader queries and send their identifiers. Being cost effective, tags are deployed on a large scale and typically used for identification of goods or even individuals. However, such a deployment comes with new security and privacy threats such as impersonation on the one hand, and tracking of tags and therewith individuals on the other hand. The cost effectiveness also implies strong limitation of computational capabilities of the tags. Current passive RFID tags can hardly afford for security mechanisms relying on complex cryptographic operations to counter the security and privacy threats.

Revisiting security problems such as authentication and privacy preserving identification in the highly constrained setting of RFID tags has given rise to a number of research activities, focusing on lightweight authentication, identification schemes, and formal security and privacy properties thereof, e.g., see Ateniese et al. [1], Bringer and Chabanne [6], Bringer et al. [7], Dimitrou [13], Pietro and Molva [21], Tsudik [25], Vaudenay [26], Weis et al. [27].

In an attempt to explore new security and privacy problems with RFID tags, we introduce a new application scenario, raising new requirements beyond the classical authentication and identification issues. The target scenario is the collection of statistics over private *properties* of a large population of individuals. Due to RFID users' demands and due to regulatory matters, the main challenge in this scenario is to *preserve the privacy* of these individuals with respect to their properties.

Addressing this scenario with RFID tags, each tag would contain the attributes of its holder in an encrypted form. The ultimate goal would be to allow a centralized party, such as a server, to compute global statistics. For example, the distribution over the properties held by a group of individuals might be of interest, but without disclosing the attributes of individuals to any party involved in the collection of these statistics.

Hence, we suggest a scheme called *PPS* ("Privacy-Preserving Statistics") that assures privacy of individual attributes in this scenario. In PPS, intermediate parties called *readers* collect encrypted properties from *tags*, compute aggregates over encrypted readings without decrypting them, and periodically forward the result of such aggregation operations to the *back-end server*. The server is then able to compute a global aggregate in cleartext based on the aggregates of the encrypted readings transmitted by readers.

The main challenge in this scenario is to allow the readers to perform the aggregation over encrypted attribute values from which the server can derive global statistics in cleartext. We address this problem through homomorphic encryption. Another threat to the privacy of the tag holders is the tracing of tags by readers. In order to circumvent this threat, we use re-encryption mechanisms. However, the scarcity of resources in tags prohibits the assignment of complex operations to tags. Therefore, the readers will perform re-encryption of the ciphertexts stored on the tags. Thus, tags do not have to perform any cryptographic operations.

The **major contributions** of PPS are:

- contrary to related work on privacy preserving tag identification, such as Ateniese et al. [1], PPS provides an RFID-based mechanism to collect statistics over a set of properties in a privacy-preserving manner.

- formal proofs of *privacy* and *unlinkability* against external eavesdroppers, malicious readers, and curious back-end servers.

- minimal hardware requirements resulting in cheap tags: PPS does not require tags to do *any cryptographic computation*, tags are passive, i.e., battery-less and only require data storage functions. Contrary to related work, PPS' storage-only requirements enable implementations on today's available EPC class1 Gen2 tags.

- data integrity: tampering with data stored on tags can be detected.

The sequel of this paper is organized as follows. In Section 2, we present a typical scenario for our application, we state the problem, and we derive the requirements for the solution. In Section 3, we present the building blocks of PPS. In Section 4, we define the notion of privacy and unlinkability in the context of our application, and we present the adversary model. Section 5 gives the formal analysis of the protocol. Finally, related work is presented in Section 6.

2. PROBLEM STATEMENT

In this section, we introduce a typical scenario for PPS, and we present a system model and the requirements PPS should fulfill.

2.1 Application scenario

The solution we propose targets applications involving a central organization that wants to collect statistics on a given population. This population will be equipped with RFID tags that will be read by readers managed by intermediary entities that are independent from the central organization.

We can imagine a scenario where the ministry of culture wants to come up with statistics about the attendance of cultural events in order to properly determine its funding policy. Basically, the cultural venues as well as the ministry of culture are interested in knowing which kind of exhibitions attract more people and which part of the population are more interested in their activity.

To that effect, the ministry will deploy readers at the entry of venues where cultural events take place such as cinemas, theaters, museums, etc. In this scenario, each potential attendee of cultural events is equipped with an RFID tag. The tag could be embedded into a museum membership card that allow the holder of the tag to get discounts on museum's events.

There could be other scenarios where a shopping mall wants to compare the activity of the shops it hosts or which type of clients it attracts the most throughout the year. The shopping mall will deploy readers at the cashiers of the shops it hosts. The visitors of the shopping mall are provided with RFID tags that could be embedded in the mall's loyalty cards that allows the visitors of the mall to get discounts on the mall services such as: parking, restaurants, etc.

To give an incentive to the visitors of museums or shopping malls to carry their tags, we embed RFID tags into membership cards/loyalty cards that allow tag owners to have discounts or privileged access to the events organized by museums or shopping malls.

Such an incentive requires binding the RFID tag to the identity of the tag owner to ensure that the RFID tag is being used by its owner when scanned. Cheap RFID tags that are the target technology of PPS *cannot* afford cryptographic authentication. Therefore, we use an out of band authentication such as a printed picture on the membership card or the loyalty card to verify the identity of the tag holder at payment for instance.

Associating the RFID tag only with the picture of its owner assures that the holder of the tag will be the actual owner while preserving his anonymity with respect to the museum or the mall he is visiting.

The RFID tag encodes the private properties of the tag holder, for example gender, age, profession etc. When the tag holder enters the venue of a cultural event for instance, the encrypted properties on the tag will be scanned by a reader. Each reader will aggregate the encrypted data it collects during a period such as a day. At the end of each period, each reader will forward the aggregate data to a server managed by the ministry of culture. The server will then update the overall statistics based on the aggregate values sent by all the readers.

The *key requirement* in these scenarios is preserving privacy: a solution should allow the server to compute global statistics over private properties of visitors while assuring the privacy of individual properties with respect to the readers and the server.

2.2 System model

PPS is the solution we propose to collect privacy preserving statistics. A typical application scenario for PPS involves several parties as follows:

- **Issuer** I: the issuer initializes each tag by writing into the tag's memory an encrypted representation of the properties of the tag holder.

- **Tags** $\{T_i\}$: each tag stores an encrypted representation of the properties of the tag holder. The encrypted representation of p properties in each tag consists of $\{P_i,\ 1 \leq i \leq p\}$ where P_i is set to "true" if the tag holder possesses the corresponding property.

- **Readers** $\{R_i\}$: readers are in charge of collecting properties stored on tags. They read the data stored on each tag and forward the result of these readings to the back-end server.

- **A back-end server** S: S processes the aggregate data received from readers and derives some global statistics such as distribution of attendance rate with respect to event types and population characteristics.

The issuer and the back-end server can be managed by independent parties. For instance, in a typical scenario, social security could act as an issuer, whereas the back-end server would be managed by the ministry of culture.

2.3 Requirements: Privacy & Unlinkability

The basic requirement for S is to count the number of tag holders satisfying each property P_i for all the properties. The main concern is to gather statistics such as counts about each property P_i, while preserving the *privacy* of tag holders. Neither readers nor the back-end server should be able to disclose the values of a tag holder's properties. To

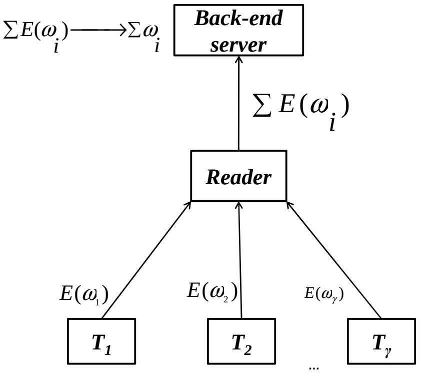

Figure 1: *Aggregation in an RFID-based system*

ensure privacy in our scheme, we propose a solution that combines encryption and aggregation. In that solution, the list ω_i of the tag holder properties are encrypted as $E(\omega_i)$ and stored on the tag. Through subsequent readings of tags in its range, the reader computes the aggregate of the ciphertexts received from the tags, $\sum E(\omega_i)$, and periodically forwards the encrypted aggregate value to the back-end server as shown in Fig. 1.

The back-end server S is the only entity that can decrypt ciphertexts. To enforce privacy against S, readers must aggregate the ciphertexts received from the tags in their range before forwarding the encrypted data to S. If the readers forward data without aggregation to the back-end server S, the latter can always tell which properties the tag holders satisfy. Nonetheless, forwarding each individual reading to the server would strongly overload typically embedded, low capacity readers.

Even though the privacy of properties is assured through encryption, *unlinkability* of tags, as defined by Chatmon et al. [10], has to be assured, too. An adversary should never be able to link two responses of the same tag over different sessions. In order to assure unlinkability, the encrypted property values sent by the same tag should be different for each reading. Re-encryption is used to that effect. In Section 4.2, we formally define the notion of unlinkability.

Note that, at first glance, PPS appears to be similar to voting, cf., Benaloh and Tuinstra [3], Sako and Kilian [23]. Instead of counting the number of tag holders satisfying a property P_i , we count the number of voters voted for a given candidate. However, the RFID settings considered in this paper are very constrained regarding a secure voting application. In secure voting, the voters are required to perform additional complex operations such as public key encryption and zero knowledge proofs to ensure not only privacy and correctness of the votes, but also other sophisticated properties such as *receipt freeness*, and *universal verifiability*, cf., Baudron et al. [2], Benaloh and Tuinstra [3], Cramer et al. [12], Sako and Kilian [23]. Clearly, these operations cannot be performed by read/write only tags.

3. PPS

Plain encryption of the properties of the tag holders ensures privacy of the data sent to readers. However, encryption prevents aggregation. Conversely, if readers decrypt the data sent by the tag at every reading, the privacy of the tag holder against readers would not be assured. Hence, we suggest to use a homomorphic encryption scheme in order to allow the aggregation of encrypted data without decryption. Homomorphic encryption allows the back-end server to derive the value $\sum \omega_i$ in cleartext from the aggregate of encrypted values $\sum E(\omega_i)$. Furthermore, aggregation is used as privacy enforcement mechanism against the back-end server in order to prevent the back-end server from deriving individual properties of a tag holder.

Even though the privacy of properties is met through homomorphic encryption and aggregation of encrypted readings, these two mechanisms do not ensure the unlinkability of tags. Unlinkability of tags is required in order to prevent the readers or eavesdropping adversaries from tracking tags over different sessions. A basic solution for unlinkability can be provided through re-encryption, cf., Golle et al. [17]. Re-encryption cannot be performed by tags, as they are completely passive, therefore, it will be performed by readers. The readers on the other hand, should not be able to decrypt the ciphertexts they receive, otherwise, they can always learn the properties a tag holder satisfies. To tackle this problem, we use an asymmetric encryption that is homomorphic.

As a well studied homomorphic asymmetric encryption scheme, Elgamal [15] meets the requirements of our application, and we use it as the underlying technique. In addition to its homomorphism, Elgamal supports re-encryption. The target scenario for our application calls for an additive homomorphism. However, Elgamal is multiplicatively homomorphic and thus falls short of suiting the target application. To cope with this limitation, the last component of the solution is a special property encoding technique based on *Gödel encoding* [16].

3.1 Elgamal Cryptosystem

- **Setup:** the system outputs two large prime P and Q such that Q divides $(P-1)$ and $|P| = \tau$. Here, τ represents the security parameter of Elgamal. Let $\mathcal{G}$ be a subgroup of $\mathbb{Z}_{\mathrm{P}}^{*}$ of order Q, and g be a generator of $\mathcal{G}$. All arithmetic operations will be performed mod P.
- **Key generation:** the secret key sk is $x \in \mathbb{Z}_{\mathrm{Q}}$. The public key pk is $y = g^x$.
- **Encryption:** to encrypt a message $m \in \mathcal{G}$, one randomly selects $r \in \mathbb{Z}_{\mathrm{Q}}$ and computes $(u, v) = (g^r, y^r m)$. The ciphertext is $c = (u, v)$.
- **Decryption:** to decrypt a ciphertext $c = (u, v)$, one computes $m = \frac{v}{u^x}$.

Elgamal encryption is multiplicatively homomorphic:

$$\forall m_1, m_2 \in \mathcal{G}, E(m_1) \cdot E(m_2) = E(m_1 \cdot m_2)$$

To adapt Elgamal to our scheme, we encode the properties using Gödel encoding before encryption as follows.

3.2 Gödel Property Encoding

In order to collect statistics on p properties P_i, we assign to each property a prime number p_i. Without loss of generality the first prime number p_1 will correspond to the property P_1, the second prime number p_2 will correspond to P_2 and so on. Both, properties P_i and primes p_i are publicly known. If the holder of a tag satisfies two properties P_i, P_j this will be represented by $\{p_i p_j\}$. More formally:

- **Setup:** let P_i, $1 \leq i \leq p$, be the p properties the back-end server is interested in, and p_i are p primes. Each property P_i will be mapped to prime number p_i.

- **Encoding:** let m be the vector $(\nu_1, ..., \nu_p)$ such that $\nu_i = 1$, if the tag T fulfills the property P_i, otherwise $\nu_i = 0$. The encoding of the properties of the tag T is defined as $\Omega(m) = \prod_{i=1}^{p} p_i^{\nu_i}$.

3.3 Protocol

In PPS, the tags are initialized once by the issuer. Whenever a tag T is read by a reader R, the reader aggregates the ciphertext $c = (u, v)$ it receives from T, then it re-encrypts the ciphertext c and writes the new ciphertext into T. Periodically, readers in the system forward their aggregates to the back-end server. The latter decrypts and decodes the aggregates and computes the statistics it is interested in.

We assume that the system comprises, for ease of understanding, a single reader, and it has γ tags in its range.

- **System setup:** let $\mathcal{G}$ be a group in which the discrete logarithm is intractable, g a generator of $\mathcal{G}$, Q the order of $\mathcal{G}$ and P_i the p properties of the system. The output of the setup operation is a pair of keys (pk, sk): $(y = g^x, x)$, $x \in \mathbb{Z}_\mathrm{Q}$, and p primes p_i such that the property P_i corresponds to prime number p_i. Elgamal secret key $sk = x$ is known by both the issuer and the back-end server. Generator g, the public key $pk = y$ and the p primes are made public.

- **Tag initialization:** the input comprises the vector $m = (\nu_1, ..., \nu_p)$, the public key y, the p primes p_i, and a random number $r \in \mathbb{Z}_\mathrm{Q}$. The issuer of the tag encodes the vector m following the Gödel encoding and computes $\omega = \Omega(m)$. The output of the initialization operation is a ciphertext $(u, v) = (g^r, y^r \omega)$.

- **Aggregation:** the input is a set of γ ciphertexts (u_i, v_i), $1 \leq i \leq \gamma$, received by the reader from the tags in its range. The reader outputs the *aggregate*, a new ciphertext $(U, V) = (\prod_{i=1}^{\gamma} u_i, \prod_{i=1}^{\gamma} v_i)$, cf., Fig. 2.

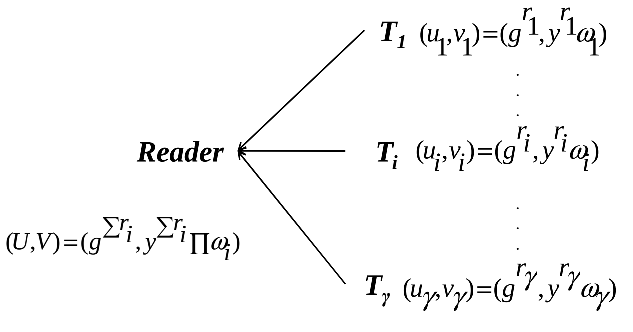

Figure 2: *Readers aggregate ciphertexts from different tags*

- **Re-encryption:** the input of re-encryption is a ciphertext $(u, v) = (g^r, y^r \omega)$ received by the reader from a tag T, g the generator of $\mathcal{G}$, the public key y, and a random number $r' \in \mathbb{Z}_\mathrm{Q}$. The output is a new ciphertext $(u', v') = (g^{(r+r')}, y^{(r+r')}\omega)$, cf., Fig. 3.

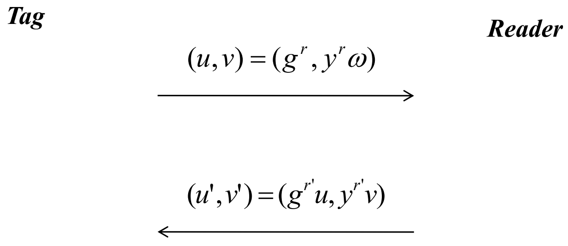

Figure 3: *Readers re-encrypt ciphertexts received from tags*

On a side note, the value $y^{(r+r')}\omega = 0 \bmod P$ is considered as "forbidden". When a reader reads a tag that stores 0, it discards the tag. This means that the reader does not aggregate or re-encrypt the tag, and the reader considers the tag as corrupted. Writing 0 into a tag is a *malicious writing* attack, see Section 4.3.

- **Decryption and decoding:** the input is a ciphertext $(U, V) = (\prod_{i=1}^{\gamma} u_i, \prod_{i=1}^{\gamma} v_i)$ received from the reader, the secret key x, and the p primes p_i. The back-end server computes $W = \frac{V}{U^x}$ and factorizes W. This factorization is easily feasible, as the back-end server knows the primes p_i. Given that this factorization is unique, the back-end server gets $\Omega^{-1}(W) = (\nu_1, ..., \nu_p)$. The respective ν_i corresponds to the number of tags satisfying the property P_i that have been read by the reader.

To get the total number of tags satisfying a property P_i in the case of **multiple readers**, the back-end server sums the ν_i for all the readers in the system.

3.3.1 *Aggregation under restrictions:*

In order to ensure the correctness of statistics obtained by the back-end server, we cannot allow the readers to aggregate an infinite number of ciphertexts. They are only allowed to aggregate up to a threshold γ of ciphertexts $c_i = E(\omega_i)$ at a time, such that $\prod_{i=1}^{\gamma} \omega_i < P$.

3.3.2 *Evaluation:*

Typically, $|P| = 1024$ bits and $|Q| = 160$ bits.

Given p properties P_i and p prime numbers p_i, the threshold γ could be defined as $\frac{|P|}{\log_2(\prod_{i=1}^{p} p_i)}$. If a reader has σ tags in its range, it will aggregate ciphertexts by bunches of size at most γ. Instead of forwarding one aggregate, the reader forwards $\lfloor \frac{\sigma}{\gamma} \rfloor + 1$ aggregates to the back-end server.

Furthermore, if the readers send to the back-end server the number of tags they read, we can reduce the number of prime numbers used in the Gödel encoding to represent the different properties, cf. Table 1. This applies in the case we have complementary properties, for instance, $(P_1, P_2) = (male, female)$. Given the total number of tags read and the number of tag holders satisfying the property P_1, we deduce the number of tag holders satisfying the property P_2. Using

Table 1: Potential properties and their encoding on museum cards

Properties	Gödel encoding
Male	2
under 25	3
Student	5
Employee	7
European union citizen	11
Disabled	13
Aggregate size γ	68

this fact leads to a more efficient property encoding and thus a larger aggregate size γ which improves the privacy of PPS against the back-end server as discussed in Section 5.2.2.

4. ADVERSARY & PRIVACY MODELS

In this section, we introduce the adversary model and define the notions of privacy and unlinkability for the proposed application.

4.1 Adversary model

PPS protects against two different categories of adversaries,

1. $\mathcal{ADV}_1$, external adversaries and malicious readers,
2. $\mathcal{ADV}_2$, a malicious back-end server.

$\mathcal{ADV}_1$ does not collude with $\mathcal{ADV}_2$.

4.1.1 $\mathcal{ADV}_1$:

Borrowing notions from Cramer and Damgård [11], we assume a *rushing, active* adversary who has full control over all communication between tags and readers. He can not only eavesdrop messages, but also intercept, modify, and even initiate communication. For example, the adversary might impersonate a tag and communicate with the reader or read-out tags. He might even replace a tag's content by re-writing it. However, re-writing tags has some special implications on PPS's security and privacy, so we discuss this issue separately in Section 4.3. Finally, the adversary might compromise readers, read-out and tamper with their memory and program – consequently, malicious readers might not behave in protocol compliant manner.

4.1.2 $\mathcal{ADV}_2$:

The back-end server might be under the control of the adversary, e.g., as assumed if the organization collecting the statistics is generally not trusted. The back-end server is not assumed to have full control over the network. The back-end server is passive in the sense that it only receives aggregates from readers. It cannot initiate communication with tags or readers.

We conjecture that there might be scenarios where back-end servers have full control over all communication and might collude with compromised readers, e.g., envisioning an extreme scenario whereby the ministry of culture would also control the readers of all the cultural venues. We clearly state that PPS will not provide privacy in such scenarios.

As motivated in the introduction, the adversary's primary goal in any case, i.e., $\mathcal{ADV}_1$ or $\mathcal{ADV}_2$, is to gain some knowledge about sensitive information, in this case individual tag holders' properties as formalized in the following privacy models.

4.2 Privacy Models

PPS borrows privacy models for storage-only tags as originally proposed by Ateniese et al. [1] and Juels et al. [20].

At the end of the protocol execution, PPS is said to be privacy-preserving, if $\mathcal{ADV}_1$ and $\mathcal{ADV}_2$

- cannot decide which properties a given tag (and therewith tag holders) satisfies.
- cannot link tags (and therewith tag holders) to previous protocol executions.

We use experiment-based definitions to formalize RFID privacy, cf., Juels and Weis [19]. In conclusion, the adversary should not have higher chance in breaking privacy or unlinkability than simple guessing. The following oracle-like constructions exist:

$\mathcal{O}_{\text{pick}}$ is an oracle that randomly selects some tags from all the n tags in the system.

$\mathcal{O}_{\text{semantic}}$ represents the oracle of semantic security of Elgamal: $\mathcal{O}_{\text{semantic}}$ is provided with two plaintexts ω_0, ω_1, randomly chooses $b \in \{0,1\}$, encrypts ω_b using Elgamal and public key pk, and returns the resulting ciphertext c_b.

$\mathcal{O}_{\text{semantic}-\text{re}}$ stands for the oracle in semantic security of Elgamal under re-encryption: $\mathcal{O}_{\text{semantic-re}}$ is provided with two Elgamal ciphertexts c_0, c_1, randomly chooses $b \in \{0,1\}$, re-encrypts c_b using public key pk, and returns the resulting ciphertext c'_b.

$\mathcal{O}_{\text{re}-\text{encrypt}}$ is an Elgamal re-encryption oracle that uses public key pk and ciphertext $c = (u, v)$ stored on tag T, and writes a new (re-encrypted) ciphertext $c' = (u', v')$ into T, cf., Section 3.3.

$\mathcal{O}_{\text{flip}}$ is an oracle that, provided with two tags T_0, T_1, randomly chooses $b \in \{0,1\}$ and re-encrypts the ciphertext stored on T_b using $\mathcal{O}_{\text{re}-\text{encrypt}}$. It returns T_b with the re-encrypted ciphertext.

$\mathcal{O}_{\text{aggregate}}$ computes a total of s aggregates Agg_1, Agg_2, …, Agg_s, each time by randomly choosing a set of γ tags, as follows: Agg_1 is computed using tags $(T_1^1, T_1^2, \ldots, T_1^\gamma)$, Agg_2 is computed using $(T_2^1, T_2^2, \ldots, T_2^\gamma)$, …, Agg_s is computed using $(T_s^1, T_s^2, \ldots, T_s^\gamma)$. The sets of tags are chosen randomly, but there is at least one tag that is an element of two different sets, i.e., used in the computation of two different aggregates. Finally, $\mathcal{O}_{\text{aggregate}}$ returns Agg_1, Agg_2, …, Agg_s.

4.2.1 Privacy against $\mathcal{ADV}_1$:

An adversary breaks the privacy of PPS, if given the public key pk, a tag T, the ciphertext $c = (u, v)$ stored on the tag T, and a property P_i, he can decide if a tag T satisfies the property P_i or not.

More formally, for τ the security parameter of Elgamal and $s \in \mathbb{N}$, we define the following privacy experiment:

Experiment $\text{Exp}^{\text{privacy}}_{\mathcal{ADV}_1}[\tau, r, s, t]$

1. **Setup:** the issuer initializes n tags with their corresponding ciphertexts using Gödel encoding and Elgamal. It publishes the public key pk. It shares its secret key sk only with the back-end server.
2. **Learning:** adversary $\mathcal{ADV}_1$ is provided with: 1.) a challenge tag $T_{\text{challenge}}$, 2.) using $\mathcal{O}_{\text{pick}}$, a total of $r-1$

different tags $\{T_i, 1 \leq i \leq r-1\}$, 3.) for each tag T_i, the list of properties P_j that T_i satisfies. Now, $\mathcal{ADV}_1$ is allowed to *read* from and *write* into T_i for a maximum of s times. $\mathcal{ADV}_1$ is allowed as well to *read* from $T_{\text{challenge}}$ for a maximum of t times. Yet, after each read or write access to any of the r tags, $\mathcal{O}_{\text{re-encrypt}}$ re-encrypts the content stored on tags.

3. **Challenge:** Given the public key *pk*, the readings' results, and the ciphertext stored on $T_{\text{challenge}}$, $\mathcal{ADV}_1$ outputs 1 if he guesses that $T_{\text{challenge}}$ satisfies P_i, and 0 otherwise. $\mathcal{ADV}_1$ *succeeds*, if his guess is right.

Definition 1. *Let $\tau \in \mathbb{N}$ be a security parameter. We consider as negligible in τ any function $\mu : \mathbb{N} \rightarrow [0, 1]$ such that $\forall c > 0, \mu(\tau) < \frac{1}{\tau^c}$ for every sufficiently large τ.*

Definition 2. *PPS is said to be privacy-preserving with respect to $\mathcal{ADV}_1$:*
if for all adversaries of category $\mathcal{ADV}_1$,

$$\Pr[\text{Exp}^{\text{privacy}}_{\mathcal{ADV}_1}[\tau, r, s, t] \text{ succeeds}] \leq \frac{1}{2} + \mu(\tau),$$

such that $\mu(\tau)$ is a function negligible in τ.

4.2.2 Privacy against $\mathcal{ADV}_2$:

Formalizing properties' privacy with respect to $\mathcal{ADV}_2$ is difficult: as assumed in the adversary model above, $\mathcal{ADV}_2$, i.e., a malicious back-end server, only receives aggregates from readers. In any case, there is no relation between tags, and therewith tag holders, and $\mathcal{ADV}_2$. In conclusion, $\mathcal{ADV}_2$ simply cannot learn anything about properties of tags.

While we do not target a formal proof, privacy against $\mathcal{ADV}_2$ is furthermore discussed and additional reasoning is given in the according security analysis section 5.1.2.

4.2.3 Unlinkability against $\mathcal{ADV}_1$:

The tags targeted in this paper are passive in that they only features storage capabilities. Hence, tags cannot update the content of their memory themselves after a read and therefore the content of a tag's memory does not change between two protocol executions. In the face of an overwhelmingly powerful adversary who can eavesdrop all communications between tags and readers, using our scheme tags would be trivially linkable. However, we conjecture that it is fair to assume that an adversary in the real world cannot continuously monitor tags and that there is at least one protocol execution ,i.e., re-encryption that is "un-observed" by the adversary. Once a tag T is re-encrypted, the adversary should not be able to link the previous interactions he has seen to tag T. In accordance with related work: *insubvertible encryption* by Ateniese et al. [1], *backward security* by Dimitrou [13] and *privacy against anonymizers* by Sadeghi et al. [22], we assume that there is at least one protocol execution that takes place outside the range of the adversary.

Under this assumption, neither external adversaries nor readers should be able to link two responses from the same tag once it is re-encrypted outside their range.

Experiment $\text{Exp}^{\text{unlinkability}}_{\mathcal{ADV}_1}[\tau, r, s, t]$

1. **Setup:** the issuer initializes n tags with their corresponding ciphertexts using Gödel encoding and Elgamal cryptosystem, it publishes its public key *pk*. It shares its secret key *sk* only with the back-end server.

2. **Learning:** the oracle $\mathcal{O}_{\text{pick}}$ provides the adversary $\mathcal{ADV}_1$ with r tags in the system that he is allowed to *read* from and *write* into for a maximum of s times. The tags are re-encrypted after each read by $\mathcal{O}_{\text{re-encrypt}}$.

3. **Challenge:** $\mathcal{ADV}_1$ is provided with two challenge tags T_0, T_1 that he is allowed to *write* into and *read* from for a maximum of t times and they are re-encrypted each time by $\mathcal{O}_{\text{re-encrypt}}$.

   Then, $\mathcal{O}_{\text{flip}}$ is queried with T_0 and T_1, $\mathcal{O}_{\text{flip}}$ provides $\mathcal{ADV}_1$ with a re-encrypted T_b. Given the public key *pk*, the results of the readings performed, and the current ciphertext stored on the tag T_b, the adversary $\mathcal{ADV}_1$ guesses the value of $b \in \{0, 1\}$. He succeeds, if his guess is right.

Definition 3. *PPS is said to provide unlinkability with respect to $\mathcal{ADV}_1$:*
if for all adversaries of category $\mathcal{ADV}_1$,

$$\Pr[\text{Exp}^{\text{unlinkability}}_{\mathcal{ADV}_1}[\tau, r, s, t] \text{ succeeds}] \leq \frac{1}{2} + \mu(\tau),$$

such that $\mu(\tau)$ is a function negligible in τ.

4.2.4 Unlinkability against $\mathcal{ADV}_2$:

A malicious back-end server should not be able to link aggregates to aggregates it has received before. More precisely, a malicious back-end server should not tell, whether a received aggregate involves a tag that was involved in another aggregate received earlier. We illustrate the unlinkability against a malicious back-end server ($\mathcal{ADV}_2$) by the following experiment:

Experiment $\text{Exp}^{\text{unlinkability}}_{\mathcal{ADV}_2}[\gamma, s]$

1. **Setup:** the issuer initializes n tags with their corresponding ciphertexts using Gödel encoding and Elgamal cryptosystem, it publishes its public key *pk*. It shares its secret key *sk* with the back-end server, i.e., $\mathcal{ADV}_2$.

2. **Learning:** $\mathcal{O}_{\text{aggregate}}$ provides $\mathcal{ADV}_2$ with s aggregates $\text{Agg}_1, \ldots, \text{Agg}_s$.

3. **Guess:** given private key *sk* and aggregates $\text{Agg}_1, \ldots, \text{Agg}_s$, $\mathcal{ADV}_2$ guesses a pair $b, b' \in \{1, \ldots, s\}$ and therewith Agg_b and $\text{Agg}_{b'}$. $\mathcal{ADV}_2$ succeeds, if Agg_b and $\text{Agg}_{b'}$ have been computed by $\mathcal{O}_{\text{aggregate}}$ with at least one tag in both aggregates.

Definition 4. *PPS is said to provide unlinkability with respect to $\mathcal{ADV}_2$:*
if for all adversaries of category $\mathcal{ADV}_2$,

$$\Pr[\text{Exp}^{\text{unlinkability}}_{\mathcal{ADV}_2}[\gamma, s] \textit{ succeeds}] \leq \frac{1}{s(s-1)} + \mu(\gamma),$$

such that $\mu(\gamma)$ is a function negligible in γ.

Untraceability. Note that in this work, we do not focus on untraceability, although this is also being considered in related work. However, the notion of untraceability is *weaker* than unlinkability, cf., Chatmon et al. [10]. Thus, as PPS provides unlinkability, it also provides untraceability.

4.3 Malicious writing

Tags in our scheme have a writable memory, where the ciphertext is stored every time it is re-encrypted by readers. As there is no access control on tags to check the authenticity of readers, our scheme is vulnerable to "malicious writing".

We can divide malicious writing attacks into two categories:

- **Writing an invalid ciphertext ("garbage") into the tag:** this attack can be detected at the back-end server, as decryption and Gödel decoding will not succeed. Moreover, if the adversary writes the value 0 into the tag, this will be detected at the next honest reader to read the tag.
- **Writing a valid ciphertext into the tag:** a malicious reader could try to alter statistics. The simplest way to implement such an attack is by copying the content of a tag into another one ("cloning"). Moreover, given that Elgamal is malleable and that the adversary knows the primes, the adversary can generate a set of valid ciphertexts from a ciphertext he has seen [14]. Since the ciphertext written into the tag is a valid one, this type of attack cannot directly be detected at decryption, and we will tackle it in the following.

Malicious writing affects the correctness of the results obtained at the back-end server. Given that access control is not feasible in our read-write only tags, this attack *cannot* be prevented. However, we propose the following solution to *detect* ciphertexts written by an adversary at the back-end server.

Instead of one ciphertext, each tag stores two ciphertexts (c, c_{ID}). The first ciphertext c encrypts the properties of the tag holder as described in the previous section. The second ciphertext c_{ID} encrypts a unique ID of the tag using standard Elgamal encryption. After a tag is scanned by a reader, the reader re-encrypts both ciphertexts c and c_{ID} and writes the new ciphertexts into the tag. The reader aggregates c and keeps a record of c_{ID}. During decryption at the back-end server, if the back-end server suspects that a received aggregate is not correct, he contacts a "trusted third party". This trusted third party (TTP) checks the records c_{ID} stored at the readers. TTP decrypts these ciphertexts and gets the IDs of the tags that were scanned along with the corresponding properties of their holders. In this manner, the TTP detects tag cloning as the ID of the cloned tag will be repeated several times.

Furthermore, in order to detect tag tampering, the tag issuer should keep a database of the tag IDs and their corresponding properties and reveal it to the TTP. Therewith, the TTP can compare the decrypted properties and the actual properties stored in the issuer database. If there is a discrepancy between the properties corresponding to the same tag ID, the TTP reports a fraud. Meanwhile, the TTP does not reveal the records of the IDs stored on the readers either to the back-end server or to the readers.

We clearly acknowledge that readers can fake statistics. However, the readers could be provided with an incentive to encourage them to ensure the integrity of the results obtained at the back-end server and therefore not to tamper with tags' content. For instance, the statistics could be used by the cultural venues or the shops to define their marketing strategy. As mentioned before, PPS focuses on privacy and unlinkability of tag holders.

5. PRIVACY ANALYSIS

This section provides formal proofs for PPS's privacy and unlinkability as defined in the models of Section 4.2.

5.1 Privacy

5.1.1 Privacy against $\mathcal{ADV}_1$:

THEOREM 1. *PPS is privacy-preserving with respect to $\mathcal{ADV}_1$ under the DDH assumption over $\mathcal{G}$.*

PROOF. Assume we have an adversary $\mathcal{A} \in \mathcal{ADV}_1$ whose advantage to break the privacy experiment is not negligible. We construct a new adversary $\mathcal{A}'$ that executes $\mathcal{A}$ as a subroutine and breaks the semantic security of Elgamal which leads to a contradiction under the DDH assumption. In this proof, we make use of the fact that a tag T satisfies a property P_j, **iff** the corresponding prime number p_j divides the plaintext underlying the ciphertext stored on T.

- $\mathcal{A}'$ picks p properties $p_i, 1 \leq i \leq p$ that he maps to p distinct primes $p_i, 1 \leq i \leq p$.
- $\mathcal{A}'$ initializes n tags as follows: he computes n Gödel encodings $\omega_j, 1 \leq j \leq n$ using the primes p_i. Provided with Elgamal public key pk, he encrypts $\omega_j, 1 \leq j \leq n$ and gets n ciphertexts that he stores on the tags.
- $\mathcal{A}'$ specifies two plaintexts $\omega_0 = \prod p_i^{\nu_{0,i}} \leq P-1$ and $\omega_1 = \prod p_i^{\nu_{1,i}} \leq P-1$, such that $\forall i$, $1 \leq i \leq p$, and $b' \in \{0,1\}$: $\nu_{b',i} \in \{0,1\}$ and $\nu_{0,i}+\nu_{1,i} = 1$. In terms of properties P_i, this means that tag T_0, storing plaintext ω_0, and tag T_1, storing ω_1, do not have a property in common.

  The adversary $\mathcal{A}'$ should specify ω_0 and ω_1 such that $\nu_{0,i} + \nu_{1,i} = 1$. Otherwise, $\mathcal{A}$ could choose a challenge property P_i that both ω_0 and ω_1 encode. In this case, the output of $\mathcal{A}$ about P_i will not provide the necessary information to $\mathcal{A}'$ to break the semantic security of Elgamal. The same holds if $\mathcal{A}$ chooses a property P_i that neither ω_0 nor ω_1 encode.
- $\mathcal{A}'$ transmits $\{\omega_0, \omega_1\}$ to the oracle $\mathcal{O}_{\text{semantic}}$.
- $\mathcal{O}_{\text{semantic}}$ returns the encryption c_b of one of the plaintexts ω_0, ω_1 to $\mathcal{A}'$.
- $\mathcal{A}'$ writes c_b into a challenge tag $T_{\text{challenge}}$.
- $\mathcal{A}'$ calls the adversary $\mathcal{A}$ that enters the learning phase. Simulating $\mathcal{O}_{\text{pick}}$, $\mathcal{A}'$ provides $\mathcal{A}$ with $r-1$ tags along with the list of properties they are satisfying. $\mathcal{A}$ is allowed to read and write into these tags for a maximum of s times. $\mathcal{A}'$ provides $\mathcal{A}$ as well with the challenge tag $T_{\text{challenge}}$. $\mathcal{A}$ has only read access to $T_{\text{challenge}}$ and he is allowed to read it for a maximum of t times. Tags are required to be re-encrypted by $\mathcal{O}_{\text{re-encrypt}}$ after being read or written into. As pk is public, $\mathcal{A}'$ can simulate successfully $\mathcal{O}_{\text{re-encrypt}}$
- $\mathcal{A}$ selects a property P_i and outputs 1 if the tag satisfies the P_i and 0 otherwise.

If $\mathcal{A}$ outputs 1, this implies that the prime number p_i corresponding to P_i divides ω_b. By construction, ω_0 and ω_1 do not have any prime divisor in common, and therefore, ω_b is the plaintext dividable by p_i.

If $\mathcal{A}$ outputs 0, this implies that p_i does not divide ω_b and by construction p_i divides ω_{1-b}. Therefore, ω_b is the plaintext that is not dividable by p_i.

$\mathcal{A}'$ can tell which plaintext ω_b corresponds to c_b. This breaks the semantic security of Elgamal ensured under the DDH assumption [24], which leads to a contradiction.

□

5.1.2 *Privacy against $\mathcal{ADV}_2$:*

As stated in Section 4.1, $\mathcal{ADV}_2$ receives only aggregated ciphertexts. Still, given the aggregates, $\mathcal{ADV}_2$ can learn some information about the properties of tags read by readers, but is never able to tell *which* tag, and therewith *which* holder satisfies *which* property.

For instance, if $\mathcal{ADV}_2$ receives an encrypted aggregate from a reader R, and decrypts it to $\mathrm{Agg} = \prod_{i=1}^{p} p_i^{\nu_i}$, and $\exists j$ such that $\nu_j = 0$ after factorization, $\mathcal{ADV}_2$ can learn that all the tags that were read by R do not satisfy the property P_j.

However, as $\mathcal{ADV}_1$ and $\mathcal{ADV}_2$ do not collude, $\mathcal{ADV}_2$ cannot tell *which* tag satisfies or does not satisfy a certain property P_i.

5.2 Unlinkability

5.2.1 *Unlinkability against $\mathcal{ADV}_1$:*

THEOREM 2. *PPS provides tag unlinkability against $\mathcal{ADV}_1$ under the DDH assumption over $\mathcal{G}$.*

PROOF. Assume we have an adversary $\mathcal{A} \in \mathcal{ADV}_1$ whose advantage to break the unlinkability experiment is not negligible. We construct a new adversary $\mathcal{A}'$ that executes $\mathcal{A}$ and breaks the semantic security under re-encryption of Elgamal.

The semantic security property of Elgamal encryption can be extended to the semantic security of Elgamal under re-encryption [20]. Let $\mathcal{A}'$ be an adversary that chooses two ciphertexts c_0 and c_1, $\mathcal{A}'$ then sends $\{c_0, c_1\}$ to $\mathcal{O}_{\mathrm{semantic-re}}$. $\mathcal{O}_{\mathrm{semantic-re}}$ flips a coin b, re-encrypts c_b to c'_b and returns c'_b to $\mathcal{A}'$. The semantic security of Elgamal under re-encryption entails that guessing the value of b is as difficult as DDH, see Juels et al. [20].

- $\mathcal{A}'$ picks p properties $p_i, 1 \le i \le p$ that he maps to p distinct primes $p_i, 1 \le i \le p$. Then, he initializes n tags.
- $\mathcal{A}'$ calls the adversary $\mathcal{A}$ that enters the learning phase. $\mathcal{A}'$ simulates $\mathcal{O}_{\mathrm{pick}}$ and provides $\mathcal{A}$ with r tags. $\mathcal{A}$ is allowed to read and write into these tags for a maximum of s times. After each reading, $\mathcal{A}'$ simulates $\mathcal{O}_{\mathrm{re-encrypt}}$ and re-encrypts the ciphertexts, as pk is public.
- $\mathcal{A}$ enters the challenge phase: $\mathcal{A}'$ simulates $\mathcal{O}_{\mathrm{pick}}$ and submits tags T_0 and T_1 to the adversary $\mathcal{A}$. $\mathcal{A}$ writes into and reads from T_0 and T_1 for a maximum of t times. $\mathcal{A}'$ can simulate $\mathcal{O}_{\mathrm{re-encrypt}}$ successfully, as pk is public.
- $\mathcal{A}'$ reads the data stored on T_0 and T_1. Basically, the data stored on a tag is a pair (u, v) that corresponds to the encryption of $\frac{v}{u^x}$ where x is the secret key of Elgamal. Without loss of generality, let c_0 (c_1 resp.) denotes the ciphertext stored on T_0 (T_1 resp.). Then, $\mathcal{A}'$ transmits c_0 and c_1 to the oracle $\mathcal{O}_{\mathrm{semantic-re}}$.
- $\mathcal{O}_{\mathrm{semantic-re}}$ returns the result c'_b of re-encrypting one of the two ciphertexts to $\mathcal{A}'$. $\mathcal{A}'$ writes c'_b into a tag T.
- $\mathcal{A}'$ calls $\mathcal{A}$ and provides him with T, simulating $\mathcal{O}_{\mathrm{flip}}$. Then, $\mathcal{A}$ outputs his guess for the value of b.

Since $\mathcal{A}$'s advantage in the unlinkability experiment is not negligible, $\mathcal{A}$ can tell which tag corresponds to the new ciphertext c'_b. If $\mathcal{A}$ outputs 0, this means that c'_b is re-encryption of c_0, otherwise c'_b is a re-encryption of c_1. Therefore, $\mathcal{A}'$ can break the semantic security under re-encryption of Elgamal that is ensured under the DDH assumption [20], again leading to a contradiction.

□

5.2.2 *Unlinkability against $\mathcal{ADV}_2$:*

THEOREM 3. *PPS provides unlinkability of tags against $\mathcal{ADV}_2$ for large γ.*

PROOF SKETCH. An aggregate $\mathrm{Agg} = \prod_{i=1}^{p} p_i^{\nu_i}$ is called *completely blinded*, **iff** $\forall i,\ 1 \le i \le p : \nu_i > 0$. Now, given a sufficiently large γ, the aggregates received by the back-end server will be completely blinded with high probability.

Therefore, the back-end server cannot distinguish between the tags involved in the aggregates. Moreover, using a large s in the learning phase would not give the adversary $\mathcal{ADV}_2$ a greater advantage in guessing (b, b').

In the following, we compute an upper bound of the advantage ϵ of $\mathcal{ADV}_2$ in the unlinkability experiment.

Let E be the event that aggregate Agg is completely blinded, so $\forall i,\ 1 \le i \le p : \nu_i > 0$. Let γ be the number of ciphertexts participating in the aggregate, and π_i is the probability that a tag holder satisfies property P_i. Without loss of generality, we assume $\pi_1 \le \pi_2 \le \ldots \le \pi_p$.

Then, the probability that $\nu_i = 0$ is $Pr(\nu_i = 0) = (1 - \pi_i)^\gamma \le (1 - \pi_1)^\gamma$.

Let $\overline{E}$ be the complementary event of E. Therefore,

$$\begin{aligned} Pr(\overline{E}) &= Pr(\nu_1 = 0 \vee \nu_2 = 0 \ldots \vee \nu_p = 0) \le \sum_{i=1}^{p} Pr(\nu_i = 0) \\ Pr(\overline{E}) &\le \sum_{i=1}^{p} (1 - \pi_i)^\gamma \le p(1 - \pi_1)^\gamma. \end{aligned}$$

$\epsilon = Pr(\overline{E})$ is the advantage of $\mathcal{ADV}_2$ in the unlinkability experiment which is **negligible** in γ. Therefore, we say that PPS is ϵ-unlinkable against $\mathcal{ADV}_2$, such that $\epsilon \le p(1 - \pi_1)^\gamma$.

Note that the advantage of $\mathcal{ADV}_2$ heavily depends on the probability π_1. If π_1 is very small, i.e., representing a *rare* property such as being disabled, PPS cannot provide unlinkability against $\mathcal{ADV}_2$. In such a case, the back-end server can link tags to aggregates. For instance, if the back-end server sees two aggregates where the property "disabled" is satisfied, it can guess with a non negligible probability that these two aggregates have one tag in common.

□

6. RELATED WORK

Juels et al. [20] were among the first to introduce re-encryption into RFID settings. Their proposal aims at ensuring the privacy of an RFID-enabled banknote. The serial number of the banknote is encrypted and the ciphertext is stored on the tag embedded in the banknote. Each time the banknote is spent, the readers in shops or banks re-encrypt the ciphertext stored on the tag. The drawback of this scheme is that the authorized readers have access to the plaintext underlying the ciphertext which allows tracking.

Golle et al. [17] introduce *universal re-encryption.* As the name implies, universal re-encryption allows re-encryption without knowing the public key initially used to encrypt the plaintexts. Re-encryption is performed by the readers in order to prevent tracking. This protocol provides key privacy – however, it fails at preventing tracing through *malicious writing.* An adversary can write into a tag a message m and encrypt it under its public key, by doing so, the adversary can always trace the tag.

To tackle this problem, Ateniese et al. [1] propose *insubvertible encryption*, which is a universal re-encryption based on bilinear pairings. This scheme makes use of randomizable certificates – based on bilinear pairings – to prove that a ciphertext stored on a tag can only be decrypted by authorized entities. If the certificate is valid, the ciphertext stored on the tag will be re-encrypted. Otherwise, it will be discarded and replaced by a *dummy encryption.* The *insubvertible encryption* as proposed in [1] allows only privacy preserving identification but not privacy preserving statistics collection which is the focus of PPS. Also, the use of elliptic curves as the underlying group in [1] requires special message encoding to map messages to points on the elliptic curve. Yet, current efficient encoding schemes fail at preserving the homomorphism of Elgamal which is an essential property to PPS. Therefore, due to the inherent lack of homomorphism in its basic building blocks, the scheme suggested by [1] does not lend itself to an extension performing privacy preserving data aggregation at the readers.

Blundo et al. [4] propose a scheme for untraceable tags using universal re-encryption and bilinear groups. Unlike Ateniese et al. [1], their construction takes place in the symmetric bilinear setting and uses *linear encryption* as proposed by Boneh et al. [5]. As in [1], the scheme suggested in [4] requires message to point mapping and therefore, it does not preserve the homomorphism of the linear encryption.

Camenisch and Groß [8] propose an attribute encoding for anonymous credentials. The scheme allows users to prove the possession of an attribute with a given value while preserving the privacy of the users. To do so, Camenisch and Groß [8] encode the attributes using Gödel encoding combined with a Camenish-Lysyanskaya signature [9] to generate the credentials. While such an approach could be theoretically used to "emulate" privacy-preserving computation of statistics, the main drawback is the requirement for complex *interactive proofs* – infeasible in our setting with storage-only tags.

Han et al. [18] propose a scheme that allows estimating the number of tags in the vicinity of a reader without collecting the ID from each RFID tag. The main idea is to infer the number of tags by examining the number of empty and collision slots in the framed slotted Aloha protocol. Although the solution proposed in [18] enables estimating the total number of tags anonymously, it does not lend itself to collect statistics on tag properties as targeted in the paper at hand.

7. CONCLUSION

RFID systems can be used for many applications besides identification and authentication. In this paper, we introduced a new application for RFID that collects statistics over a population of tag holders. We presented PPS, a protocol to mitigate resulting new privacy problems. PPS does not require tags to perform any (cryptographic) computation. Instead, tags only need to feature some cheap storage. All computations within PPS are solely performed by readers. PPS provably ensures the *privacy* of tags and therewith holders' properties as well as their unlinkability: tag holders can be sure that neither RFID readers, nor a back-end system can reveal the properties stored on their tags. Additionally, if scanned at different readers on different occasions, tag holders can be sure that these occasions cannot be linked.

References

[1] Giuseppe Ateniese, Jan Camenisch, and Breno de Medeiros. Untraceable rfid tags via insubvertible encryption. In *CCS '05: Proceedings of the 12th ACM conference on Computer and communications security*, pages 92–101, New York, NY, USA, 2005. ACM. ISBN 1-59593-226-7.

[2] Olivier Baudron, Pierre-Alain Fouque, David Pointcheval, Jacques Stern, and Guillaume Poupard. Practical multi-candidate election system. In *PODC '01: Proceedings of the twentieth annual ACM symposium on Principles of distributed computing*, pages 274–283, New York, NY, USA, 2001. ACM. ISBN 1-58113-383-9.

[3] Josh Benaloh and Dwight Tuinstra. Receipt-free secret-ballot elections (extended abstract). In *STOC '94: Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 544–553, New York, NY, USA, 1994. ACM. ISBN 0-89791-663-8.

[4] Carlo Blundo, Angelo De Caro, and Giuseppe Persiano. Untraceable tags based on mild assumptions. Cryptology ePrint Archive, Report 2009/380, 2009. `http://eprint.iacr.org/`.

[5] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *In proceedings of CRYPTO âĂŹ04, LNCS series*, pages 41–55. Springer-Verlag, 2004.

[6] Julien Bringer and Hervé Chabanne. Trusted-HB: A low-cost version of HB $^+$ secure against man-in-the-middle attacks. *IEEE Transactions on Information Theory*, 54(9):4339–4342, 2008. ISSN 0018-9448.

[7] Julien Bringer, Hervé Chabanne, and Emmanuelle Dottax. HB^{++}: a lightweight authentication protocol secure against some attacks. In *SecPerU*, pages 28–33, 2006. ISBN 0-7695-2549-0.

[8] Jan Camenisch and Thomas Groß. Efficient attributes for anonymous credentials. In *CCS '08: Proceedings of*

the 15th ACM conference on Computer and communications security, pages 345–356, New York, NY, USA, 2008. ACM. ISBN 978-1-59593-810-7.

[9] Jan Camenisch and Anna Lysyanskaya. Security in communication networks, third international conference, scn 2002, amalfi, italy, september 11-13, 2002. revised papers. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN*, volume 2576 of *Lecture Notes in Computer Science*. Springer, 2003.

[10] Christy Chatmon, Tri van Le, and Mike Burmester. Secure anonymous RFID authentication protcols. Technical report, Florida State University, Department of Computer Science, Tallahassee, Florida, USA, 2006. `http://www.cs.fsu.edu/~burmeste/TR-060112.pdf`.

[11] Ronald Cramer and Ivan Damgård. Introduction to secure multi-party computations. In *Contemporary Cryptology: Advanced Courses in Mathematics*, pages 41–87. Birkhauser, 2005. ISBN 3-7643-7294-X.

[12] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *Advances in Cryptology âĂŤ EUROCRYPT âĂŹ97*, Lecture notes in computer science, pages 103–118. Springer-Verlag, 1997. ISBN 978-3-540-62975-7.

[13] Tasso Dimitrou. rfidDOT: RFID delegation and ownership transfer made simple. In *Proceedings of International Conference on Security and privacy in Communication Networks*, Istanbul, Turkey, 2008. ISBN 978-1-60558-241-2.

[14] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. In *STOC '91: Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 542–552, New York, NY, USA, 1991. ACM. ISBN 0-89791-397-3.

[15] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 10–18, New York, NY, USA, 1985. Springer-Verlag New York, Inc. ISBN 0-387-15658-5.

[16] Kurt Gödel. über formal unentscheidbare sätze der principia mathematica und verwandter systeme i. *Monatsheft für Mathematik und Physik*, 38:173–198, 1931. ISSN 0026-9255.

[17] Philippe Golle, Markus Jakobsson, Ari Juels, and Paul Syverson. Universal re-encryption for mixnets. In *In Proceedings of the 2004 RSA Conference, Cryptographer's track*, pages 163–178. Springer-Verlag, 2002.

[18] Hao Han, Bo Sheng, Chiu C. Tan, Qun Li, Weizhen Mao, and Sanglu Lu. Counting RFID tags efficiently and anonymously. In *Proceeding of IEEE Infocom*, San Diego, USA, 3 2010.

[19] Ari Juels and Stephen A. Weis. Defining strong privacy for RFID. In *PERCOMW '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 342–347, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-2788-4.

[20] Ari Juels, Ravikanth Pappu, and Thingmagic Llc. Squealing euros: Privacy protection in rfid-enabled banknotes. In *Financial Cryptography âĂŹ03*, pages 103–121. Springer-Verlag, 2002. ISBN 978-3-540-40663-1.

[21] Roberto Di Pietro and Refik Molva. Information confinement, privacy, and security in RFID systems. In *Lecture Notes in Computer Science, Volume 4734*, pages 187–202, 2007. ISBN 978-3-540-74834-2.

[22] Ahmad-Reza Sadeghi, Ivan Visconti, and Christian Wachsmann. Anonymizer-Enabled Security and Privacy for RFID. In *8th International Conference on Cryptology And Network Security – CANS'09*, Kanazawa, Ishikawa, Japan, December 2009. Springer. ISBN 978-3-642-10432-9.

[23] Kazue Sako and Joe Kilian. Secure voting using partially compatible homomorphisms. In *CRYPTO '94: Proceedings of the 14th Annual International Cryptology Conference on Advances in Cryptology*, pages 411–424, London, UK, 1994. Springer-Verlag. ISBN 3-540-58333-5.

[24] Yiannis Tsiounis and Moti Yung. On the security of elgamal based encryption. In *PKC'98, LNCS 1431*, pages 117–134. Springer-Verlag, 1998. ISBN 978-3-540-64693-8.

[25] Gene Tsudik. YA-TRAP: yet another trivial RFID authentication protocol. In *Proceedings of International Conference on Pervasive Computing and Communications Workshops*, Pisa, Italy, 2006. ISBN 0-7695-2520-2.

[26] Serge Vaudenay. On privacy models for RFID. In *Advances in Cryptology - Asiacrypt 2007*, Lecture Notes in Computer Science, pages 68–87. Springer-Verlag, 2007. ISBN 978-3-540-76899-9.

[27] S.A. Weis, S.E. Sarma, R.L. Rivest, and D.W. Engels. Security and privacy aspects of low-cost radio frequency identification systems. In *Security in Pervasive Computing*, pages 201–212, Boppard, Germany, 2003. ISBN 3-540-20887-9.