Delphine Christin, Christian Roßkopf, Matthias Hollick, Leonardo A. Martucci, Salil S. Kanhere: *IncogniSense: An Anonymity-preserving Reputation Framework for Participatory Sensing Applications*. In: Proceedings of the 10th International Conference on Pervasive Computing and Communications (PerCom), pp. 135-143, 2012.

IncogniSense: An Anonymity-preserving Reputation Framework for Participatory Sensing Applications

Delphine Christin*, Christian Roßkopf*, Matthias Hollick*, Leonardo A. Martucci†, Salil S. Kanhere‡
* Secure Mobile Networking Lab, Technische Universität Darmstadt, Darmstadt, Germany
† Telecooperation Lab, Technische Universität Darmstadt, Darmstadt, Germany
‡ School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
Emails: firstname.lastname@cased.de, salilk@cse.unsw.edu.au

***Abstract*—Reputation systems rate the contributions to participatory sensing campaigns from each user by associating a reputation score. The reputation scores are used to weed out incorrect sensor readings. However, an adversary can de-anonmyize the users even when they use pseudonyms by linking the reputation scores associated with multiple contributions. Since the contributed readings are usually annotated with spatiotemporal information, this poses a serious breach of privacy for the users. In this paper, we address this privacy threat by proposing a framework called IncogniSense. Our system utilizes periodic pseudonyms generated using blind signature and relies on reputation transfer between these pseudonyms. The reputation transfer process has an inherent trade-off between anonymity protection and loss in reputation. We investigate by means of extensive simulations several reputation cloaking schemes that address this tradeoff in different ways. Our system is robust against reputation corruption and a prototype implementation demonstrates that the associated overheads are minimal.**

I. Introduction

Recent mobile phones are equipped with a plethora of embedded sensors, integrate widespread wireless technologies and complex processing capabilities. These technological features have contributed to the emergence of a new paradigm known as *participatory sensing* [1]. Participatory sensing applications involve volunteers collecting sensor readings from the surrounding environment using their mobile phones. The collected sensor readings are reported to application servers, where summaries are computed and published in the form of maps and statistics. Several practical systems based on this novel paradigm have been developed in recent years, which include creating noise pollution maps [2] and obtaining real-time road traffic information [3].

Participatory sensing applications are however exposed to incorrect contributions due to their inherent open nature [4]. For example, participants may inadvertently position the phone in an undesirable position while collecting sensor readings (e.g., phone kept in bag while sampling street-level noise). Moreover, malicious participants may deliberately contribute bad data. Both behaviors result in erroneous contributions, which need to be identified and eliminated to ensure the reliability of the computed summaries. For this purpose, reputation systems tailored to participatory sensing have been proposed such as [4]. They assign reputation scores to the participants based on the quality of their contributions and then use these scores to weed out bad contributions. The reputation scores can also serve as incentives for the users to participate in the application. Such systems however need to observe the contributions made by each device for an extended period of time to compute the reputation and hence, require linkability across multiple contributions from the same device. An adversary can exploit these links to de-anonymize the volunteers and compromise their privacy, since the sensor readings usually include spatiotemporal meta-data [5]. In this paper, we propose a solution that addresses this inherent conflict between privacy and reputation requirements. Our specific contributions can be summarized as follows:

1) We present IncogniSense, an anonymity-preserving reputation framework based on blind signatures [6], which is agnostic to both the reputation assignment algorithm and the application. In IncogniSense, each user picks a new pseudonym for each time period, which is used to report sensor readings. Before the next period starts, the user transfers the reputation score associated with his current pseudonym to his next pseudonym. This allows the user to conserve his reputation across multiple periods, while limiting associations between his contributions to a unique period.
2) IncogniSense cloaks the reputation to be transferred to prevent an attacker from linking multiple pseudonyms. As reputation cloaking has an inherent trade-off between anonymity protection and loss in reputation, we explore this design space by undertaking a detailed simulation-based analysis of several cloaking mechanisms and we especially study their resilience against linking attacks.
3) We conduct a thorough threat analysis showing the robustness of IncogniSense against reputation corruption.
4) We evaluate the feasibility of our approach by implementing a proof-of-concept on Android Nexus S mobile phones. Assuming that a new pseudonym is chosen every 5 minutes, IncogniSense only incurs an additional 2.3% of energy expenditure, which is a small price to pay for the enhanced privacy protection offered.

The paper is organized as follows. In Section II, we discuss related work. We introduce our threat model in Section III and present the IncogniSense framework in Section IV. We conduct a multi-dimensional evaluation of IncogniSense in Section V, before making concluding remarks in Section VI.

II. Related Work

We compare IncogniSense with existing anonymity-preserving reputation systems designed for application domains orthogonal to participatory sensing, as this specific problem has not been addressed in the context of participatory sensing, other than a mention of its importance as future research [7]. To the best of our knowledge, we are the first to propose a concrete framework for this paradigm including a proof-of-concept implementation and a multi-dimensional evaluation. While this paper discusses our solution in the context of participatory sensing, IncogniSense has generic applicability and is not necessarily restricted to resource-constrained devices such as mobile phones.

Manifold reputation architectures based on pseudonyms have been proposed for peer-to-peer networks. For example, in [8], pseudonyms are created by Trusted Platform Modules and attested by Certificate Authorities. In the solution presented in [9], the users utilize a set of pseudonyms with the same reputation to collect electronic coins centralized at a third party. In contrast to our solution, both approaches are vulnerable to identity-based attacks since users can control an arbitrary number of pseudonyms. Another scheme presented in [10] proposes context-based, self-certified, and Sybil-free pseudonyms built using e-tokens. The users, however, cannot change their pseudonyms once they are associated to a context. Finally, STARS [11] extends existing peer-to-peer reputation systems by traceability and anonymity. Complete anonymity of honest clients is however not guaranteed since the scheme reveals the identity of clients suspected to corrupt their reputation and is subject to false positives.

IncogniSense shares the most similarities with [12] and [13], which rely on periodic pseudonyms and transfer of reputation between pseudonyms. As our work builds upon the RuP algorithm [12], we present a detailed comparison in Section IV-D. In short, RuP requires the clients to create n temporary pseudonyms for each valid pseudonym and relies on probabilistic proofs. As such, a malicious client can create multiple pseudonyms and tamper with its reputation with a success probability of $1/n$. Increasing n reduces the threats to reputation manipulation, but increases associated overheads. In comparison to RuP, IncogniSense is robust against reputation corruption and the clients generate only one pseudonym per period. The protocol presented in [13], which extends the ideas from [12], is based on the concept of k-anonymity [14]. They assume the existence of a third party server which forms groups of clients sharing the same reputation and distributes a signature key per group to sign the pseudonyms. The clients must however trust each other not to collude with the third party and/or other clients to reduce the anonymity set. Moreover, the proposed protocol introduces additional overhead for the clients to create pseudonyms as compared to [12] and our solution.

III. Threat Model and Assumptions

In this section, we present our threat model and detail our assumptions. We consider that our adversary set includes malicious clients, application servers, and the *reputation and pseudonym manager* (RPM) (described in Section IV). The adversaries follow the Dolev-Yao threat model [15], i.e., they are able to listen to all communication, fabricate, replay, and destroy messages. They are, however, not able to break cryptographic mechanisms.

A. Threats to Reputation

The goal of the adversaries, primarily malicious clients, is to corrupt the reputation system in order to artificially increase their own reputation. Self-promotion can be achieved by means of Sybil attacks, in which the malicious client creates an arbitrary number of identities that vouch for each other [16]. Alternatively, malicious clients may replay old messages to gain reputation without contributing new sensor readings.

We assume that the application server and the RPM are protected against fraudulent access by well-established security mechanisms. Hence, adversaries are not able to access stored data, or change the behavior of applications. Denial of service attacks are considered out of the scope of this paper.

B. Threats to Anonymity

Another goal of the adversaries is to breach the anonymity of the clients. They track the interactions of the clients with the reputation system and attempt to establish relationships between successive pseudonyms and link them to a unique real identity. We assume that the reported sensor readings do not contain any direct indication of the identity of the clients and that the interactions of the clients with the application server are anonymized using, e.g., disposable IP and MAC addresses and anonymous communication networks [17].

IV. The IncogniSense Framework

We first provide an overview of the IncogniSense framework and detail the underlying mechanisms. We then present our reputation cloaking mechanisms and highlight the differences between IncogniSense and the framework proposed in [12], which forms the underlying basis for parts of our work.

A. Overview

The IncogniSense framework illustrated in Fig. 1 includes clients, an application server, and a RPM (introduced in Section III). The clients repeatedly collect sensor readings and report them to the application server using pseudonyms, which have a pre-determined validity period T. Each client reports its sensor readings using its current pseudonym $P_{current}$ generated in association with the RPM and based on blind RSA signatures [6]. Blind signatures ensure the authenticity of signed messages without revealing their content to the signing entity and prevent the signing entity from linking the message content with the identity of its creator. By employing blind signatures, the RPM can guarantee that each client has a unique pseudonym for each T. This not only prevents Sybil attacks but also makes it impossible to link the pseudonym to the real identity and thus protects the anonymity of the client. The application server runs a reputation algorithm, which

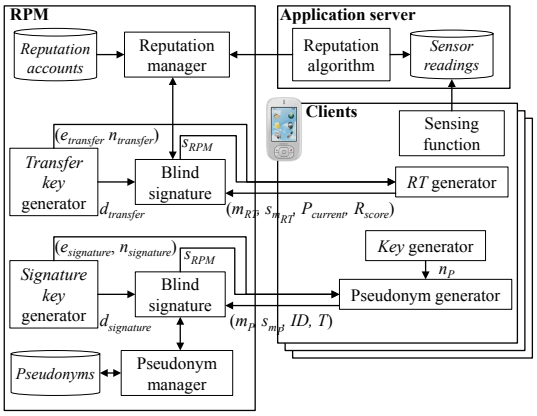


Fig. 1. IncogniSense framework

evaluates the sensor readings after each reporting and attributes a reputation score R_{score} to $P_{current}$. The reputation score is then transferred by the application server to the RPM, which maintains reputation accounts for each pseudonym. Before the end of T, the clients generate their next pseudonym P_{next} and transfers R_{score} from $P_{current}$ to P_{next} to conserve the acquired reputation score after the expiration of $P_{current}$. The reputation transfer is realized using a *reputation token* RT generated by the clients in collaboration with the RPM to prevent reputation corruption. The transfer process also makes use of blind signatures to prevent the RPM from linking $P_{current}$ to P_{next}. The application server eventually makes use of the reputation scores to identify inaccurate contributions. Depending on the selected reputation model, the application server can discard the contributions with low reputation scores or reduce the weighting for such contributions in the computation of summaries. Note that the design of both reputation algorithm and model is considered out of the scope of this paper. Existing solutions such as [4] can however be easily integrated into our generic framework.

B. Underlying Mechanisms

The proposed solution is comprised of the following four steps, which are repeated sequentially for each time period T. We assume that each client has a permanent identifier ID, a private key (d_{client}, n_{client}), a public key (e_{client}, n_{client}), and is registered with the RPM. We also assume that the RPM generates a set of *transfer keys* in the bootstrapping phase.

1) Pseudonym Generation: For each T, the RPM generates a new private/public pair of *signature keys* common to all clients: $(d_{signature}, n_{signature})$ and $(e_{signature}, n_{signature})$. The client first generates a private/public key pair (d_P, n_P) and (e_P, n_P) for its new pseudonym. The client uses n_P as its new pseudonym referred to as P and generates the corresponding signature s_P as follows. The client first prepares the message m_P using n_P, the public *signature key*, and the blinding factor r modulo n_P, as follows:

$$m_P = n_P \cdot r^{e_{signature}} \bmod n_{signature} \tag{1}$$

The client creates a signature s_{m_P} using a function of the concatenation f of the triplet (m_P, ID, T) signed with its permanent private key to guarantee the authenticity of m_P:

$$s_{m_P} = f(m_P \parallel ID \parallel T)^{d_{client}} \bmod n_{client} \tag{2}$$

The client transmits m_P, s_{m_P}, its ID, and the interval of validity T for P to the RPM for blind signature. The RPM verifies the authenticity of m_P and that the client has no existing pseudonym for this interval. After verification, the RPM generates the blind signature s_{RPM} signing m_P:

$$s_{RPM} = {m_P}^{d_{signature}} \bmod n_{signature} \tag{3}$$

The client finally generates the pseudonym's signature s_P from the blind signature s_{RPM} that achieves the generation of P, which becomes $P_{current}$:

$$s_P = s_{RPM} \cdot r^{-1} \bmod n_{signature} \tag{4}$$

2) Reporting of Sensor Readings: Within T, the client periodically reports sensor readings to the application server using $P_{current}$, the reporting period being common to all clients. The application server verifies the validity of $P_{current}$ with the RPM, evaluates the sensor reading using a reputation model, and attributes a reputation score R_{score} to $P_{current}$, with $R_{score} \in \mathbb{Z}$. The application server transmits R_{score} to the RPM, which add R_{score} to $P_{current}$'s reputation account.

3) Generation of Reputation Tokens: Before the expiration of $P_{current}$ (i.e., at the end of T), the client generates P_{next} for the next T (see step 1). It then requests the value of $P_{current}$'s reputation account from the RPM and uses one or several reputation tokens to transfer it to P_{next}. The generation of each *RT* is comparable to the generation of the pseudonym, except that the client does not generate any key pair and both client and RPM use the aforementioned *transfer keys* for the signature of the messages. Each *transfer key* pair is associated to a reputation value and determines the reputation associated to a given *RT*.

For each *RT*, the client selects a random bit string ID_{RT} as identifier and prepares the message m_{RT} for blind signature using the public *transfer key* corresponding to the *RT*'s value:

$$m_{RT} = ID_{RT} \cdot r^{e_{transfer}} \bmod n_{transfer} \tag{5}$$

The client signs the message m_{RT} with the signature $s_{m_{RT}}$ using the private key $(d_{P_{current}}, n_{P_{current}})$ associated to $P_{current}$. The real identity of the client is hence not revealed while transferring reputation from one pseudonym to the next.

$$s_{m_{RT}} = f(m_{RT} \parallel P_{current} \parallel R_{score})^{d_{P_{current}}} \bmod n_{P_{current}} \tag{6}$$

The client transmits m_{RT}, $s_{m_{RT}}$, $P_{current}$, and R_{score} to the RPM for blind signature. The RPM verifies that m_{RT} is used for the first time as well as the balance of the reputation account of $P_{current}$ before decrementing it by R_{score}. After verification, the RPM blindly signs m_{RT} with the corresponding private *signature key*. The client finally uses the blind signature to generate the final signature of the *RT*.

4) Reputation Transfer: Within the current T, the client registers with the RPM using P_{next} and hands over *RT*(s). The RPM verifies that each *RT* has not been used before and credits P_{next}'s reputation account from the *RT*'s value.

C. Cloaking Mechanisms

In theory, the utilization of blind signatures prevents the linking of consecutive pseudonyms. However, in practice, the reputation transfer between two consecutive pseudonyms may reveal insights about their succession. For example, consider the case where $P_{current}$ has the highest reputation among all pseudonyms. Now assume that after reputation transfer, this same reputation score is associated with the pseudonym, P_{next}. It is fairly straightforward for an adversary to establish a link between $P_{current}$ and P_{next}. An adversary able to track the reputation scores over several time intervals can link pseudonyms used in different periods. To prevent such attacks, we propose that the clients cloak their reputation scores before their transfer. Since the RPM prevents unjustified reputation promotion by controlling the generation of the *RT*s, the clients only transfer reputation scores lower or equal to their actual reputation. While cloaking the reputation scores prevents a reputation analysis attack, it may cause degradation in reputation score since it entails adding a perturbation to the same. In this section, we present three different reputation cloaking schemes, which address this tradeoff in different ways. We evaluate their performances in Section V-B.

1) Floor Function Reputation Transfer (Floor): This scheme divides the entire spectrum of reputation scores into fixed intervals and classifies the clients into these intervals based on their reputation scores. For the actual transfer, the clients use the floor value of the reputation interval as reputation score and transfer it using a single *RT*. Note that we use the floor value to prevent unjustified reputation promotion. Similar to the concepts of k-anonymity [14], the scheme forms groups of pseudonyms sharing the same reputation score. The larger the groups, the more indistinguishable the pseudonyms, and the harder it is for the adversary to link consecutive pseudonyms. The anonymity protection is thus determined by the size of the groups, which depends on the selection of the intervals. Selecting large intervals may increase the number of pseudonyms within a group, but may negatively affect the reputation scores due to the coarse granularity of the chosen intervals. The size of the reputation intervals is therefore an important design parameter for addressing the tradeoff between anonymity and reputation.

2) Transfer of Random Sets from Reputation Partition Sets (RandSet): RandSet divides the reputation score to be transferred into multiple *RT*s based on a predefined set partitioning (e.g., (10, 50, 250)). For each reputation transfer, the division of the reputation score into the set partitioning is determined by each client individually. For example, a client can partition a reputation score of 70 into one *RT* of value 50 and two *RT*s of value 10 or 7 *RT*s of value 10. The client then randomly selects one or more *RT*s handed over to the RPM. We refer to p as the probability that an individual *RT* is used to transfer reputation. In other words, each *RT* is discarded and not transferred with a probability $1 - p$. Both the division and discarding mechanisms increase the entropy of the actually transferred reputation and prevent the linkage between consecutive pseudonyms. However, discarding *RT*s can reduce the reputation scores. The diminution in reputation is linearly correlated with the size of the *RT*s, i.e. discarding a large *RT* leads to greater loss in reputation. Applying a set partitioning with small *RT* values has a lower impact on the transferred reputation, but incurs computation and transmission (i.e., bandwidth) overheads for the clients and the RPM. In summary, both set partitioning and discarding probability influence the tradeoff between anonymity and reputation, and thus need to be analyzed.

TABLE I
COMPARISON OF THE CRYPTOGRAPHIC OVERHEAD

		RuP [12]	IncogniSense
Client	Generated key pairs	n per interval	1 per interval
	Prepared messages	n per interval	1 per interval
RPM	Generated key pairs	1	1 per interval
	Signature verifications	n-1 per interval and per client	0
	Number of signatures	1 per interval and per client	1 per interval and per client

3) Transfer of Random Scores from Reputation Partition Sets (RandScore): The initial reputation score is split into different *RT*s of predefined size as in *RandSet*. During the transfer, all the *RT*s are transferred. However, the transferred score of the individual *RT*s is lowered according to a random function. The set partitioning impacts on the entropy of the transferred scores and the loss in reputation. The range of the possible scores linearly increases with the size of the *RT*s. Simultaneously, the transferred reputation statistically diminishes, as the probability of transferring the initial *RT* values decreases, while the size of the *RT*s increases.

Note that the spectrum of possible cloaking mechanisms is not limited to the aforementioned schemes. We have specially selected these schemes based on the diversity of their key parameters to address the tradeoff between anonymity protection and reputation degradation and to provide a thorough and extensive evaluation in Section V-B.

D. Comparison with the Existing Framework

We contrast IncogniSense with the framework proposed in [12], since our work improves its ideas. In particular, we highlight weaknesses of the RuP algorithm in terms of cryptographic overhead and vulnerability to reputation manipulation and argue how IncogniSense addresses these issues.

1) Algorithmic Differences: In the RuP algorithm, the client includes the interval of validity of the pseudonym in the blind signature. Each client creates n pseudonyms to generate a sole valid pseudonym. The n pseudonyms are transmitted to the RPM, which randomly selects $n-1$ pseudonyms and requests the client to send the corresponding random values r^{-1}. The RPM then encrypts the $n-1$ pseudonyms and verifies that they are valid for the same interval. If the verification is successful, it signs the nth pseudonym, which becomes the valid pseudonym. The generation of *RT*s also necessitates that

the client prepares n messages for blind signature and the RPM verifies $n-1$ messages before signing the nth message. In IncogniSense, we decouple the interval of validity of the pseudonym and the value of the reputation to transfer from the blind signatures by introducing periodic *signature keys* and different *transfer keys* for each value of *RT*, reciprocally.

2) Cryptographic Overhead: Table I summarizes the cryptographic overhead for generating pseudonyms associated with both schemes. In [12], the client generates n key pairs, selects n values r and r^{-1}, and executes n encryptions to generate one valid pseudonym. The generated key pairs of non-selected pseudonyms should not be reused to prevent the RPM from linking the pseudonyms to the client's identity. In IncogniSense, the client prepares one blind signature per pseudonym, i.e., it generates only one key pair and encrypts one message, and the RPM verifies only one signature per interval and per client (instead of $n-1$), which significantly reduces the computational overhead if we assume a large base of clients. Similar conclusions can be drawn for the *RT* generation, except that no key pair is generated by the client.

3) Reputation Manipulation: In the RuP algorithm, the RPM cannot verify the interval of validity of the pseudonym included in the blinded message. It randomly verifies $n-1$ of the n generated pseudonyms and signs the nth, hoping that it contains the same validity interval as the verified pseudonyms. This implies that malicious clients can generate multiple pseudonyms for a given time interval with a probability of $1/n$. If such an attack is successful, then the adversaries can seriously compromise the reputation system (see Section III-A). Likewise, malicious clients can generate fraudulent *RT*s and increase their reputation. IncogniSense completely eliminates the possibility of Sybil attacks and also prevents adversaries from compromising the reputation transfer process. The utilization of periodic *signature keys* and the verification of already existing pseudonyms by the RPM guarantees that each client has a unique pseudonym per interval. Similarly, the utilization of different *transfer keys* allows the RPM to easily verify and guarantee the value of the transferred reputation, preventing malicious clients from generating falsified *RT*s.

In summary, IncogniSense achieves better protection against reputation manipulation, while significantly reducing the cryptographic overhead for the client. Note that the reputation transfer mechanisms are further compared in Section V-B.

V. Evaluations

The goal of our evaluations is threefold: (1) analyze the robustness of IncogniSense against identified threats to reputation, (2) measure the level of anonymity and quantify the reduction in reputation scores for the different cloaking schemes, and (3) empirically measure the overhead for the mobile clients.

A. Robustness against Threats to Reputation

We consider the assumptions and threat model presented in Section III and argue that IncogniSense is resilient against the following attacks.

1) Sybil Attacks: Malicious clients can attempt to generate multiple pseudonyms for a given interval to increase their reputation through cross-recommendations. However, IncogniSense is protected against these attacks since the RPM maintains a list of the clients that have presented pseudonyms for blind signature along with their validity interval. Once a pseudonym has been signed for a given interval, the RPM discards all other pseudonyms submitted by the same client.

2) Replay Attacks: Malicious clients may attempt to replay old messages for the following two reasons: (1) artificially increase their reputation by replaying *RT*s, (2) debit the reputation account of honest clients without the victims receiving the associated credit by replaying m_{RT} (see Eq. 5). The RPM, however, maintains lists of both IDs (i.e., ID_{RT} and the final ID) associated with each *RT* (see Section IV-B4). The RPM can thus detect malicious clients trying to corrupt the reputation system and hence, prevents both attacks. Malicious clients are prevented from forging *RT*s and manipulating the reputation of other clients, as the creation of the *RT*s requires the signature of the corresponding pseudonym, which is only possible using the pseudonym's private key.

3) Manipulation of Reputation Accounts: Malicious clients can attempt to alter the reputation stored in the RPM. The clients, however, do not have direct access to their reputation accounts and the RPM is protected against unauthorized access using standard cryptographic primitives. As such, this attack cannot be launched.

4) Reporting of Falsified Sensor Readings: Malicious clients can try to report falsified sensor readings on behalf of others to degrade their reputation. IncogniSense protects honest clients against this attack by requesting clients to authenticate with the application server and the RPM. These entities verify the validity of the pseudonyms before considering their contributions and/or delivering them information. Such an attack would only be successful if malicious clients can access the private keys of the targeted clients and those of their respective pseudonyms, which is beyond the scope of our attacker model.

In summary, we have shown that IncogniSense is robust by design against a variety of threats directed against the reputation system by malicious clients.

B. Anonymity Protection and Reduction in Reputation Score

We simulate the resilience of IncogniSense against the threats to anonymity identified in Section III-B. In particular, we measure the level of anonymity, i.e. unlinkability, that can be achieved by the different reputation cloaking schemes introduced in Section IV-C. As reputation cloaking has an inherent trade-off between anonymity protection and loss in reputation, we also quantify the reduction in reputation score caused by the different cloaking approaches.

1) Simulation Setup and Method: We implemented a simulator in Java to model the behavior of clients, RPM and application server. Each simulation run is for 100 time intervals. We repeat each simulation 100 times and present the averaged results. All clients remain active for the duration of

TABLE II
PROBABILITY DISTRIBUTION OF REPUTATION SCORES

Attribution probability	0.25	0.35	0.25	0.1	0.05
Reputation scores	10	5	0	-5	-10

the simulation. During each interval, the clients generate 5 random sensor readings and report them to the application server. The application server runs a simulated reputation algorithm, which randomly attributes reputation scores to the pseudonyms according to the distribution presented in Table II. Note that the actual values of both sensor readings and reputation scores do not impact the performances of the cloaking schemes in terms of linkability and can thus be selected randomly. Moreover, we consider the study of additional reputation models and distributions as future work.

We adopt the assumptions of our threat model (see Section III) and assume that the RPM and the application server are malicious internal observers. We further assume that adversaries collude and aim at linking consecutive pseudonyms. For each simulated interval, the RPM and the application server have access to the following information. The RPM observes the generated *RT*s, the utilized *RT*s, and the updates of the reputation accounts, while the application server observes the current pseudonyms, the initial and final reputation scores, and the updates of reputation scores.

Based on the observed information, the adversaries identify the set of pseudonyms active in each interval. We refer to $S_{P_{current}}$ as the set of pseudonyms active in a given interval and $S_{P_{next}}$ as the set of pseudonyms active in the subsequent interval. As all clients remain active for the duration of the simulation, $|S_{P_{current}}| = |S_{P_{next}}|$ and there exists a bijection between the sets $S_{P_{current}}$ and $S_{P_{next}}$. For each pseudonym $P_{current}$, the adversaries attempt to identify its successor P_{next}, according to the following operations. The adversaries first look for pseudonyms in $S_{P_{next}}$ that have a reputation score lower or equal to the reputation score of $P_{current}$ before the end of the current period. Indeed, the clients can have either transferred their full reputation or cloaked it, incurring loss in reputation. Then, the adversaries identify all pseudonyms in $S_{P_{current}}$ that have a single link to a pseudonym in $S_{P_{next}}$ and eliminate all other links due to the bijection between $S_{P_{current}}$ and $S_{P_{next}}$. Similarly, they identify all pseudonyms in $S_{P_{next}}$ having a single link to a pseudonym in $S_{P_{current}}$ and eliminate all other links. Finally, as a measure of the level of anonymity provided by the reputation transfer, we calculate how many potential successors x_k (i.e., links) a pseudonym k has on average. We express the amount of potential successors as the fraction of the total number of clients $N = |S_{P_{current}}|$. For each interval, we calculate this metric as follows.

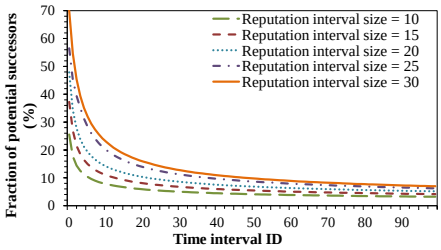
$$\frac{1}{N} \cdot \sum_{k=1}^{N} x_k \tag{7}$$

A value of 1 implies perfect anonymity achieved during the transfer, since all pseudonyms within $S_{P_{next}}$ are potential successors to each individual pseudonym of $S_{P_{current}}$. The lower the value, the smaller the set of pseudonyms that are the potential successors, which in turn implies that it may be easier to establish a link between consecutive pseudonyms. We can therefore directly assess the quality of the anonymization achieves by the cloaking schemes using this metric.

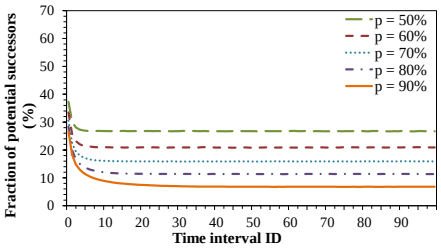
2) Evaluation Results: By applying the above setup and method, we first individually evaluate the reputation cloaking schemes (see Section IV-C). In particular, we analyze the effect of the most important configuration parameter relevant to each scheme and the resulting impact on the anonymity of the clients. Secondly, we compare how these schemes fare against each other. In this comparison, we use the transfer scheme used in [12] as a baseline. This scheme, also referred to as the *Full Reputation Transfer* (*Full*) scheme, always transfers the entire reputation in a single *RT*. It hence represents the worst case in terms of anonymity protection, since the reputation score remains the same for consecutive pseudonyms, which allows for easy linkability by the adversaries. If not noted otherwise, we investigate a population of 100 clients and use *RT*s of value 10 for easier comparison of the different schemes. Note that we assume a constant and synthetic workload to facilitate the fair comparison of the schemes. A study of dynamic user populations and workloads is considered as future work. These schemes can also be applied for larger populations, although, anonymity protection becomes increasingly important as the population size decreases.

In the *Floor* scheme, the clients use the floor value of given reputation intervals as reputation score and transfer it using a single *RT*. Fig. 2(a) illustrates the influence of different interval sizes (10, 15, 20, 25, 30) on the quality of the anonymity protection. The results show that the choice of large reputation intervals results in a higher level of protection, as more clients are grouped within the same interval and therefore become indistinguishable during the *RT* transfer. This protection comes at the expense of greater perturbation in reputation for some of the clients, since the difference between the actual reputation value and the floor of the interval also increases on average. Over time, the protection level decreases, since the reputation values of the individual clients spread over wider intervals as shown in Fig. 5(b) for an interval size of 20. Fig. 5 shows the development of reputations scores over time and is annotated with the 0.25- and 0.75-quantile of reputation scores of the clients for time interval 50. As a result, individual clients are more easily distinguishable in our attacker model, since potential successors can be identified more easily. The *Full* scheme shows a similar behavior (cf. Fig. 5(a)), but is able to preserve higher reputation scores. This is illustrated in Fig. 4, which shows the observed diminution in reputation scores per interval caused by the cloaking. With both schemes, clients are therefore easy to identify after a number of rounds since only a small number of peers have similar reputation scores.

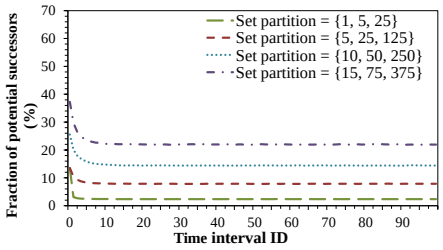
In the *RandSet* scheme, the reputation score is partitioned into different *RT*s of predefined size and a reputation transfer is performed with probability p for each *RT*. Fig. 2(b) illustrates the influence of the choice of p (50%, 60%, 70%, 80%, 90%) using the set partitioning (10, 50, 250) on the anonymity protection level. The fraction of potential successors stabilizes early, which means that attackers do not gain further evidence



(a) Floor: Fraction of potential successors over time for selected sizes of reputation interval

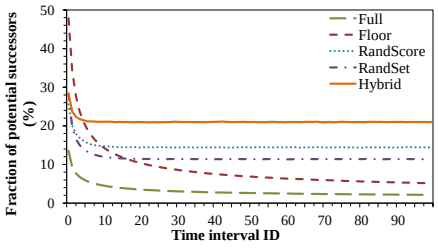


(b) RandSet: Fraction of potential successors over time for different discarding probabilities $1 - p$

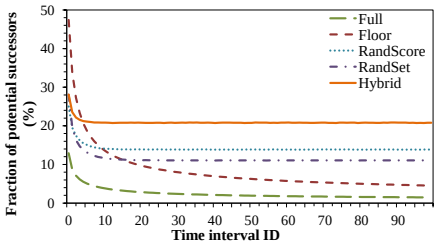


(c) RandScore: Fraction of potential successors over time for different set sizes with a fixed score discarding probability of 25%

Fig. 2. Level of unlinkability for reputation transfer schemes Floor, RandSet, and RandScore



(a) 100 clients



(b) 300 clients

Fig. 3. Scheme comparison and impact of the number of clients on the percentage of successors

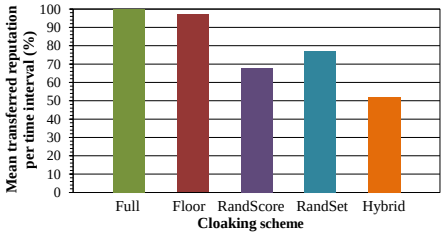
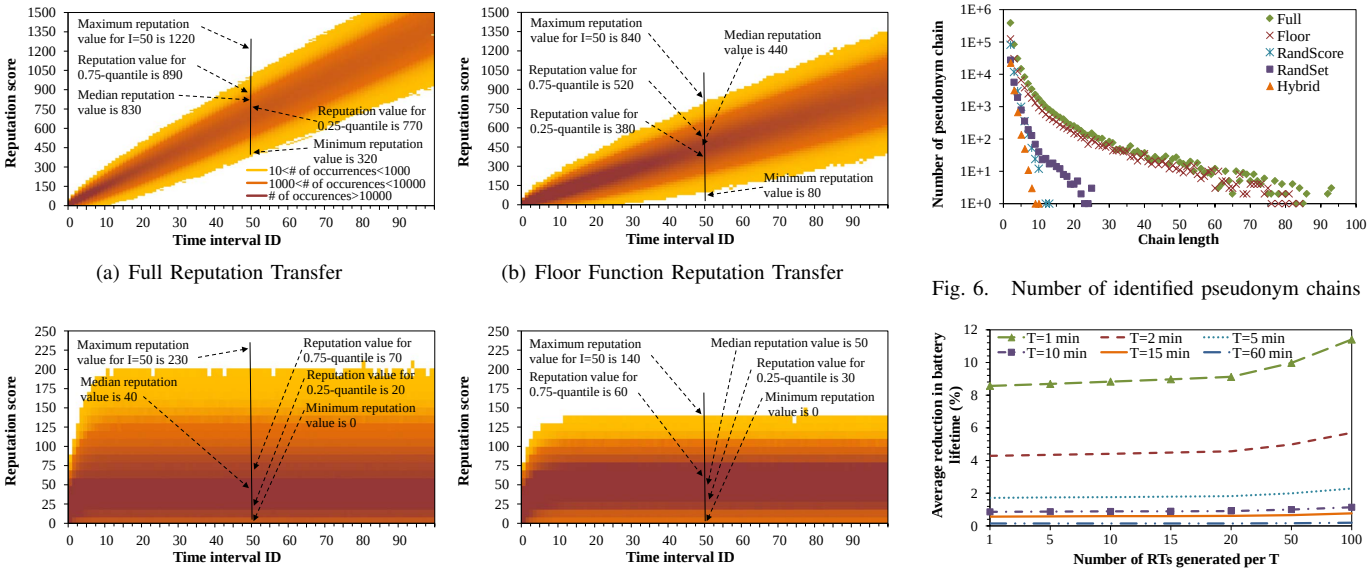


Fig. 4. Comparison of transferred reputation

even if monitoring for extensive periods of time. This is due to the reputation scores, which are not constantly increasing over time, but fluctuate in a certain interval as shown in Fig. 5(c) for a probability p of 80%. Indeed, at a certain time instant the average increase in reputation due to the contribution to the application equals the average loss due to the discarding of reputation during the transfer. In contrast, the *Full* and *Floor* schemes constantly increase the reputation values as shown in Fig. 5(a) and 5(b). The percentage of potential successors using *RandSet* lies between 7% and 26% for p equal to 90% and 50%, respectively (see Fig. 2(b)). The degree of anonymity protection increases with decreasing p i.e., increasing the discarding probability $1 - p$. Fig. 4 shows the preserved reputation score for p equal 80%. Obviously, for increasing p, the preserved reputation per round increases likewise. Note that the preserved reputation stays slightly below p, since the partitioning of the sets might not exactly fit the exact reputation value.

In the *RandScore* scheme, the reputation score is also partitioned into several *RT*s, which are all used in the reputation transfer. The transferred score is however lowered according to a random function. We analyze the influence of different set partitions while randomly discarding between 0% and 50% (following a continuous uniform distribution) of the reputation score of a *RT*. Fig. 2(c) shows that the level of anonymity protection increases with the set partitioning moving towards larger *RT* values. The smaller the lowest *RT* size in a set (1, 5, 10, 15), the worse the protection of anonymity, since individual clients can now be tracked by the number of *RT*s exchanged (but not the random reputation score exchanged). Similar to *RandSet*, the performance quickly stabilizes since the reputation values inherently stay within certain bounds as shown in Fig. 5(d) (for a partition set of (10, 50, 250) and a discarding probability of 25%), which makes identification very hard even for sophisticated attackers. Note that *RandSet* and *RandScore* perform similarly, if the same partitioning is applied and p equals the mean of the score discarding probability used in *RandScore*.

For comparison purposes, we select one variant of each scheme that demonstrated a good tradeoff between anonymity protection and reputation loss based on the previous analysis. The parameterization is as follows. *Floor* utilizes an interval size of 20; *RandScore* partitions the *RT* sets into: 10, 50, 250 and has a mean value of the discarding probability of 25%; *RandSet* retains *RT*s with a probability p of 80% and utilizes a similar partitioning (10, 50, 250). To model a participatory sensing scenario, we investigate populations of 100 and 300 clients. We present the results in Fig. 3 for all proposed reputation cloaking schemes for both scenarios under consideration. We include an additional scheme referred to as *Hybrid*, which combines the *RandSet* and *RandScore* schemes. The *Hybrid* scheme utilizes the partitioning (10, 50, 250) and retains *RT*s with a probability p of 80%. The reputation score of the retained *RT*s is then randomly lowered as in the *RandScore* scheme. The *Hybrid* scheme is robust against observers that can track the number of *RT*s or the reputation scores obtained/transferred, since it cloaks both values during transfer. As seen from both graphs in Fig. 3, all schemes are rather robust in the face of increasing the client base from 100 to 300. In summary, the *Hybrid* scheme provides the best anonymity protection followed by *RandScore*, *RandSet*, *Floor*, and the *Full* scheme. Due to the constantly increasing reputation, *Full* and *Floor* degrade over time, while the other schemes provide a constant protection.



(a) Full Reputation Transfer

(b) Floor Function Reputation Transfer

(c) Random Set Reputation Transfer

(d) Random Score from Set Reputation Transfer

Fig. 5. Distribution of reputation score amongst clients over time

Fig. 6. Number of identified pseudonym chains

Fig. 7. Estimated daily cost in battery lifetime

Fig. 6 shows the number and the length of the pseudonym chains, i.e. the amount of identified consecutive pseudonyms, for all schemes. Note that chains of length two represent identified links between two consecutive pseudonyms. Schemes that allow for long pseudonym chains are weaker than schemes that only allow the attacker to identify short chains. The *Hybrid* scheme is the most resilient scheme against attacks since it only allows adversaries to build very short chains of unique predecessor-successor relations (always shorter than 10). The *RandScore* (no chains longer than 13) and the *RandSet* (no chains longer than 25) have a vast majority of unlinkable clients, which is in line with our previous observations of a good anonymity protection. The *Floor* and the *Full* schemes perform worst in terms of identified chain length and number of identified chains. In a few cases, they allow adversaries to identify chains of up to 84 and 93 pseudonyms, respectively. Since the reputation scores with *Full* and *Floor* constantly increase, it is much easier for an attacker to follow individual pseudonyms. As a countermeasure, one could limit the maximum reputation score a client can obtain. The introduction of an upper bound would help to keep well behaving clients together in terms of reputation scores, thus making it much harder for an attacker to build pseudonym chains. However, a careful study of the effects of upper bounds is considered out of the scope of this work.

The anonymity of the clients is, however, protected at the price of a reduction in reputation as shown in Fig. 4. Clients utilizing the *Hybrid* scheme suffer the highest loss in reputation, followed by the *RandScore* and the *RandSet* schemes. The *Full* and the *Floor* schemes manage to transfer nearly the complete reputation scores.

In summary, we have demonstrated that some of the presented schemes can provide an effective anonymity protection, even if adversaries are able to track the reputation transfer between pseudonyms for long periods of time. We have shown that the cost of a high degree of anonymity protection is the loss of reputation. If we assume that all clients adopt the same cloaking scheme, all clients will be similarly affected by the incurred loss in reputation. This is consistent with the typical use of reputation in participatory sensing applications, where the application server is mainly interested in comparing the reputation scores of different clients and correspondingly associate a weight to their sensor readings. In the rare case where the absolute reputation score is of interest, the users may consider using a scheme, such as *Full* or *Floor*, which preserves the reputation value but is more susceptible to attacks. However, if minor perturbations in the scores are acceptable, then schemes such as *RandScore*, *RandSet*, and *Hybrid* may be considered. Note that our evaluation setup has been designed as the best case scenario for a potential adversary, since we assume that each client is continuously active and cannot store reputation scores for future use. In a real-world deployment, the clients may however report sensor readings intermittently. This complicates the identification process of consecutive pseudonyms for the adversaries and thus, improves the anonymity protection provided by the schemes. The performance of the cloaking schemes can be further improved by introducing reputation transfer between non-successive periods. We however consider an analysis of these additional features as future work.

C. Empirical Evaluation of Overheads

We implemented a proof-of-concept of IncogniSense to demonstrate the feasibility of our approach. In particular, we quantified the overhead in terms of energy consumption for the mobile clients. The overhead should be maintained as low as possible so as to not drain the battery of the mobile phones. In our implementation, the mobile client program was developed on Android Nexus S phones, while the application

TABLE III
MEASURED OVERHEAD PER *RT* AND KEY GENERATION FOR THE CLIENTS

	RT	Key pair
Average battery lifetime (hour)	5:09	5:18
Average number of executions	5037500	16860
Average execution time (ms)	4	1129

server and RPM were implemented as two Apache Tomcat servlets. The activities of the clients (including the cloaking schemes) are managed by a background thread. The blind RSA signatures are based on 1024-bit private/public key pairs. The pseudonyms and *RT*s are stored in a SQLite database on the clients, the access to which is strictly restricted to our application. The RPM especially maintains lists of the generated pseudonyms and utilized *RT*s in MySQL databases. The clients, the RPM, and the application server communicate via Wi-Fi and the communications are secured using HTTPS.

We conducted a first experiment to measure the impact of the generation of pseudonyms, *RT*s, and keys on the clients' battery lifetime. We configured a benchmarking client to repeatedly execute the aforementioned sequence of operations until the exhaustion of the battery. We disabled all other programs and functions and repeated each experiment 20 times on different clients. The results presented in Table III indicate that the overhead to generate keys for the new pseudonyms is significantly higher than the overhead to create *RT*s. Note that the overhead to create a pseudonym is equal to the overheads caused by both *RT* and key pair generation. IncogniSense saves 90% of energy while reducing the probability of reputation corruption from 0.1 to 0 as compared to [12] assuming $n = 10$. In a second experiment, we configured a client to generate pseudonyms and *RT*s with a period T. We examined the impact of both the duration of T and the number of generated *RT*s on the battery usage and compared it with a reference implementation. Note that the impact of the cloaking schemes on the battery usage is negligible compared to the cryptographic operations required to generate *RT*s and pseudonyms. Fig. 7 shows the overall reduction in battery lifetime per day imputed to our approach. By selecting T greater than 5 minutes, the daily cost in battery lifetime remains below 2.3%, rendering our approach especially feasible for resource-constrained devices such as mobile phones.

VI. CONCLUSIONS

We have proposed an anonymity-preserving reputation framework called IncogniSense, which is agnostic to both the applications and the applied reputation algorithm. Our system utilizes periodic pseudonyms which are generated using blind signature and relies on a secure reputation transfer mechanism between these pseudonyms. We have introduced the concept of reputation cloaking to prevent an adversary from de-anonymizing the users by linking the reputation scores associated with their contributions. As cloaking has an inherent trade-off between anonymity protection and loss in reputation, we have explored the design space and extensively examined the performances of several different schemes. Based on this analysis, we have provided guidelines for the choice of the appropriate cloaking scheme in accordance with the application requirements. We have demonstrated the resilience of our system against typical threats. Finally, a prototype implementation confirms the feasibility of our solution.

ACKNOWLEDGMENTS

The authors would like to thank Andreas Reinhardt for the fruitful discussions. This work was partially supported by CASED (www.cased.de) and EIT ICT Labs (www.ictlabs.eu).

REFERENCES

[1] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. Srivastava, "Participatory Sensing," in *Proc. of the 1st Workshop on World-Sensor-Web (WSW)*, 2006, pp. 1–5.

[2] R. Rana, C. Chou, S. S. Kanhere, N. Bulusu, and W. Hu, "Ear-Phone: An End-to-end Participatory Urban Noise Mapping System," in *Proc. of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2010, pp. 105–116.

[3] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden, "CarTel: A Distributed Mobile Sensor Computing System," in *Proc. of the 4th ACM International Conference on Embedded Networked Sensor Systems (SenSys)*, 2006, pp. 125–138.

[4] K. L. Huang, S. S. Kanhere, and W. Hu, "Are You Contributing Trustworthy Data?: The Case for a Reputation System in Participatory Sensing," in *Proc. of the 13th ACM International Conference on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWiM)*, 2010, pp. 14–22.

[5] D. Christin, A. Reinhardt, S. S. Kanhere, and M. Hollick, "A Survey on Privacy in Mobile Participatory Sensing Applications," *Journal of Systems and Software*, vol. 84, no. 11, pp. 1928–1946, 2011.

[6] D. Chaum, "Blind Signatures for Untraceable Payments," in *Advances in Cryptology: Proceedings of Crypto 82*, 1983, pp. 199–203.

[7] I. Krontiris and N. Maisonneuve, "Participatory Sensing: The Tension Between Social Translucence and Privacy," in *Trustworthy Internet*, 2011, pp. 159–170.

[8] M. Kinateder and S. Pearson, "A Privacy-Enhanced Peer-to-Peer Reputation System," in *E-Commerce and Web Technologies*, 2003, vol. 2738, pp. 206–215.

[9] E. Androulaki, S. G. Choi, S. M. Bellovin, and T. Malkin, "Reputation Systems for Anonymous Networks," in *Privacy Enhancing Technologies*, 2008, pp. 202–218.

[10] L. A. Martucci, S. Ries, and M. Mühlhäuser, "Sybil-Free Pseudonyms, Privacy and Trust: Identity Management in the Internet of Services," *Journal of Information Processing*, vol. 19, no. 1, pp. 1–15, 2011.

[11] Z. Zhang, J. Liu, and Y. Kadobayashi, "STARS: A Simple and Efficient Scheme for Providing Transparent Traceability and Anonymity to Reputation Systems," in *Proc. of the International Workshop on Autonomous and Spontaneous Security (SETOP)*, 2010, pp. 170–187.

[12] H. Miranda and L. Rodrigues, "A Framework to Provide Anonymity in Reputation Systems," in *Proc. of the 3rd Annual International Conference on Mobile and Ubiquitous Systems: Networks and Services (MobiQuitous)*, 2007, pp. 1–4.

[13] Y. Wei and Y. He, "A Pseudonym Changing-based Anonymity Protocol for P2P Reputation Systems," in *Proc. of the 1st International Workshop on Education Technology and Computer Science (ETCS)*, 2009, pp. 975 –980.

[14] L. Sweeney, "K-anonymity: A Model for Protecting Privacy," *International Journal of Uncertainty, Fuzziness, and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[15] D. Dolev and A. C. Yao, "On the Security of Public Key Protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[16] J. R. Douceur, "The Sybil Attack," in *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, 2002, pp. 251–260.

[17] M. Shin, C. Cornelius, D. Peebles, A. Kapadia, D. Kotz, and N. Triandopoulos, "AnonySense: A System for Anonymous Opportunistic Sensing," *Journal of Pervasive and Mobile Computing*, vol. 7, no. 1, pp. 16–30, 2010.