A Framework Balancing Privacy and Cooperation Incentives in User-Centric Networks

Alessandro Aldini
University of Urbino "Carlo Bo"
Urbino, Italy
email: alessandro.aldini@uniurb.it

Abstract—**User-centricity subsumes new models of Internet connectivity and resource sharing, which are based on collaborative behaviors asking for cooperation strategies. On one hand, typical incentives stimulating cooperation, based, e.g., on trust and remuneration, require some level of information disclosure that can be used to outline the user behavior. On the other hand, disclosing such information can be considered as a privacy breach keeping the users from being involved in certain interactions. In this paper, we present a flexible privacy-preserving mechanism trading privacy for trust-based and cost-based incentives. Firstly, the proposed mechanism is validated theoretically through model checking based analysis. Secondly, implementation issues are discussed with respect to the design of ad-hoc solutions based on a centralized reputation system and a distributed trust system.**

Keywords–user-centric networks; privacy; trust; cooperation incentives; model checking.

I. INTRODUCTION

Nowadays, user-driven services, like personal hotspot and peer-to-peer, play a fundamental role in the reshaping of the Internet value chain. The growing trend towards autonomic user-centric architectures is moving the focus on the user experience, related needs, expectations, and attitude to cooperation. One of the key factors behind the success of community-scale user-centric initiatives is given by the user involvement as a *prosumer*, i.e., an actor combining the roles of service producer and consumer. Such an involvement can be guaranteed only by taking into account several orthogonal aspects, including the need for incentive mechanisms stimulating the willingness to collaborate, the user perception of the trustworthiness of agents and means supporting the community infrastructure, the major issues related to information privacy and risk management arising in a framework favoring the active participation of unknown users.

In a recent work presented at SECURWARE 2014 [1], a novel approach has been proposed to set up a flexible and efficient cooperation infrastructure favoring collaborative behaviors on the basis of specific user's needs in terms of social (e.g., personal sensibility to trust and privacy issues) and economical (e.g., in terms of costs that can be afforded) requirements. The objective of this work – which is a revised and extended version of [1] partially based also on material appeared in [2] – is to show that different dimensions of the problem surveyed above, like trust, privacy, and cooperation incentives, can be effectively balanced to fulfill all the user requirements at the basis of an active involvement as a prosumer in user-centric networks.

The first fundamental aspect governing any interaction in user-centric networks is trust [3]. Establishing stable trustworthiness relations among unknown users is the objective of trust and reputation systems [4]. Trust can be viewed as the subjective belief by which an individual expects a given entity to perform with success some activity on which individual's welfare depends. Reputation emerges implicitly or explicitly in the community as an objective estimation about the level of honesty, integrity, ability, and disposition of each user as perceived by the other members of the community. It is quite natural to rely on trust and reputation information to take decisions about the opportunity to collaborate with certain partners. To this aim, several explicit mechanisms providing estimations of trust and reputation have been proposed in the literature to stimulate and guide cooperation [5], [6], among which we concentrate on those providing computational estimations of user's trustworthiness. Basically, these estimations work effectively as an incentive to collaborate if they represent parameters influencing access to services at favorable conditions, among which we include the service cost as another important aspect affecting the perceived quality of experience. In fact, remuneration is a widely used kind of incentive stimulating cooperation [7], as very often sense of community, synergy, and trust do not suffice to overcome the limitations of obstacles like, e.g., selfishness and, even worse, cheating, which represent threats keeping users from being cooperative. Whenever combined with trust, remuneration enables a virtuous circle for the proliferation of user-centric services.

On the other hand, trust is a concept that may involve and justify the disclosure of personally identifiable sensitive information, which in general can be perceived as a dramatic breach of privacy, thus playing a deterrent role when users are getting involved in interactions. In practice, the lower the attitude to expose sensitive information is, the higher the probability of being untrusted when negotiating a service. Trading privacy for trust is thus a way for balancing the subjective value of what is revealed in exchange of what is obtained [8].

These considerations motivate the need for a flexible cooperation model in the setting of user-centric networks. In the following, we first comment on related work to emphasize the kind of flexibility we would like to obtain with respect

to classical models that integrate trust and privacy. Then, in Section II we describe a novel cooperation model in which privacy is managed and traded with trust. In Section III, we analyze formally the proposed model, even through a comparison with classical ones. This is done in the setting of a real-world cooperation system for user-centric networks [9]. The aim is not only to show that a balanced tradeoff between privacy and trust can be achieved, but also to emphasize the impact of such a tradeoff upon other aspects – like the service cost – that are in some relation with trust. Formal modeling and analysis are based on automata theory and model checking [10]. In Section IV, we discuss all the implementation issues of the proposed approach. In particular, the applicability of the cooperation model is shown under two main practical scenarios. On one hand, we first discuss a centralized reputation-based approach to the implementation of the novel model of privacy management. This framework is based on the presence of a trusted third party (TTP) collecting information about every transaction completed. By combining the subjective estimations on the trust towards each user, the TTP makes them available to the community with the aim of making explicit a collective notion of reputation, while keeping the desired level of privacy for every user involved. On the other hand, we show how to implement the same model in the setting of distributed systems that cannot rely on TTP at run time. In such a case, a trust system is implemented that is based on user's personal experience and, possibly, recommendations provided by neighbors. Finally, some conclusions terminate the paper in Section V.

A. Related Work

Trust and privacy represent two pillars for any social platform aiming at offering resource and information sharing among users [11]. Trading several different cooperation incentives stimulates honest behaviors while keeping users from cheats and selfishness. For instance, it is well-known that making trust and service cost mutual dependent is a winning strategy in the setting of user-centric networks [9], [12], [13], as also proved formally by means of formal methods, like game theory and model checking [14]–[18]. Combining these aspects also with user's privacy is a challenging issue. As an example, the unavoidable contrast between privacy and trust is mitigated by the approach proposed in [19], where it is shown that these two aspects can be traded by employing a mechanism based on *pseudonyms*. In practice, users create freely pseudonyms identified by the so-called *crypto-id*, i.e., the hash of the public key of a locally generated asymmetric cryptography key pair. Then, in different environments, a user can use different pseudonyms to carry out actions logged as events signed with the private key of the chosen pseudonym. If needed to acquire more reputation, several pseudonyms can be linked together in order to augment the number of known actions and potentially increase the trust in the linked entity. However, in approaches such as this one the link is irrevocable.

Developing trust based schemes to enforce trustworthy relations in anonymity networks is another active research field (see, e.g., [20] and the references therein). Incentive mechanisms are proposed in [21] to achieve a balanced tradeoff between privacy and trust in the setting of data-centric ad-hoc networks. In [22], such an interplay is formulated as an optimization problem in which both privacy and trust are expressed as metrics. In [23], trust towards an entity is used to take decisions about the amount of sensitive information to reveal to the entity. Further works on unlinkability [24] and pseudonymity (see, e.g., [25], [26]) provide insights on the tradeoff between privacy and trust.

A typical characteristic of the approaches proposed in the literature is concerned with the incremental nature of privacy disclosure. In fact, sensitive information linking is irrevocable and, as a consequence, any privacy breach is definitive. Instead, the approach proposed in this work aims at relaxing such a condition.

We conclude the state-of-the-art presentation by citing two practical systems that deal with privacy and trust management in cooperative networks. Identity Mixer [27] allows users to control and minimize the amount of personal data they have to reveal in any access request. By selectively disclosing only the information strictly needed for access, different transactions performed by the same user become unlinkable, thus avoiding tracking of users. U-Prove [28] provides a cryptographic platform allowing users to minimally disclose certified information during transactions. In particular, user credentials are generated dynamically and encode only the attributes chosen by the user in a way that makes different transactions unlinkable. In both systems, differently from our proposal, unlinkability is a semantic notion depending on the specific attributes involved in the transactions. Moreover, no computational notion of trust is explicitly employed.

As a specific contribution of our approach that makes it different with respect to other methodologies, a collective notion of trust is employed that ensures privacy through identity obfuscation and offers incentive mechanisms stimulating honest, collaborative behaviors in user-centric networks.

II. Modeling Privacy Management

In a classical view of privacy, a user exposes (part of) personal information in order to be trusted enough to get access to the service of interest. In other words, privacy disclosure is traded for the amount of reputation that the user may need to be considered as a trustworthy partner in some kind of negotiation in which, e.g., service cost may depend on trust. Typically, once different pieces of sensitive information, say I_1 and I_2 (which may represent credentials, virtual identities, or simply the proof of being the user involved in a transaction previously conducted), are linked and exposed to be trusted by someone else, then such a link is irrevocably released. In this view, we say that the disclosure of sensitive information is *incremental* along time.

In order to exemplify, as discussed in [19], I_1 and I_2 may identify two different transactions conducted by the user under two different pseudonyms, each one revealing different personal user data. The user is obviously able to show that both I_1 and I_2 are associated with the same origin and, if such a proof is provided, I_1 and I_2 become irrevocably linked together.

As opposite to the scenario discussed above, we envision an alternative model of privacy release in which the link is not definitive. This is achieved if, for each new transaction conducted by the user, the amount of privacy disclosure is *independent* of the information released in previous interactions. Such a flexibility would allow the user to tune the amount of information to disclose in order to negotiate a transaction at the desired level of privacy without taking care of previous and future interactions. With respect to the example above, we intend that once I_1 and I_2 are linked to complete a given transaction, in a future interaction the same user can decide to break such a connection and expose, e.g., only I_1, with the guarantee that I_2 will be not associated with such an interaction.

In order to make it possible such a revocation mechanism, the idea consists of introducing some form of uncertainty associated with the owners of specific actions. Let us explain how to achieve such a condition by employing the virtual identity framework of [19]. As mentioned above, a virtual identity is represented by the crypto-id. The basic idea of the independent model of privacy release is that trust and transactions are mapped to pieces of the crypto-id, called *chunks*, rather than to the crypto-id as a whole.

Consider, e.g., a typical handshake between Alice, who issues a service request, and Bob, who offers the service. Instead of revealing to be Alice, she accompanies the request with a portion of her crypto-id identified by applying a bitmask to the crypto-id through the bitwise AND operation. Therefore, a chunk is a subset of bits of the crypto-id, of which we know value and position. Amount and position of 1's occurrences in the bitmask are under Alice's control.

The transaction is then identified by the chunk chosen by Alice. Hence, trust values (and related variations due to the feedback following the transaction execution) are not associated with Alice directly, but are related to the chunk of bits extracted from Alice's crypto-id through the chosen bitmask. In general, the same chunk is potentially shared by other crypto-ids belonging to several different users. In other interactions, Alice may select different chunks of her crypto-id. Moreover, she can also spend a set of chunks of her crypto-id in order to exploit a combination of the trust associated with each of these chunks. Thanks to the uncertainty relating chunks and associated owners, every time Alice exposes a chunk to Bob in order to negotiate a transaction, Bob cannot link the current transaction to any of the previous transactions conducted (by Alice or by other users) by using the same chunk or one of its possible subsets or supersets.

Example 1. *For the sake of presentation, consider a 8-bit crypto-id, e.g.,* 10010101*, and calculate the chunk revealing the 2nd and 5th bits of the crypto-id. This is obtained through the following bitwise operation:*

	10010101	*(crypto-id)*
AND	00010010	*(bitmask)*
=	000$\underline{1}$00$\underline{0}$0	*(chunk)*

Notice that the same bitmask identifies the same chunk if applied to the crypto-id 00011100*.*

In the following, we say that a crypto-id K matches a given chunk C if there exists a bitmask that, applied to K via the bitwise AND operation, returns C (in this case, we sometimes say also that C matches K). If two crypto-ids K_1 and K_2 coincide for the bit values identified by a certain bitmask, then they both match the resulting chunk. In other words, whenever the two users identified by K_1 and K_2 use such a chunk, then they are indistinguishable from the viewpoint of the other members of the community. When necessary, we use the extended notation C_B to identify a chunk resulting from the application of bitmask B and the usual vector based notation $C_B[i]$ (resp., $B[i]$) to denote the value of the i-th bit of the chunk (resp., bitmask).

In practice, the chunk sharing principle discussed above represents the basic mechanism enabling the form of identity obfuscation needed by the independent model of privacy release. Strictly speaking, the uncertainty relating chunks and owners is not granted absolutely, as it may happen that a chunk identifies univocally a crypto-id, especially whenever the population is small. However, several solutions can be applied to manage the transient phase during which the community is growing, e.g., by injecting fictitious crypto-ids until the critical mass is reached. Hence, we can safely assume that a deterministic matching between chunk and crypto-id is statistically irrelevant.

An important effect of chunk sharing concerns trust and reputation management. In fact, the trust $t(C)$ towards a chunk C represents a collective notion of the trust towards the set S of users with crypto-id matching C. Hence, the approximation with which $t(C)$ represents the actual trustworthiness of the user employing C in the current transaction depends on the size (and composition) of S. Calculating correct trust estimations by just knowing chunks that can identify several different users is just one of the critical aspects. Another one is concerned with the validation of the chunk exposed by the user in a transaction, who is expected to prove to be a proper owner of the chunk without actually revealing the related crypto-id.

While all these practical issues are discussed in Section IV, in the next section we abstract away from any implementation detail and we ask whether, in general, an independent model of privacy release is worth to be considered with respect to classical, incremental disclosure models. As we will see, an answer to such a question can be provided by applying model checking based formal methods.

III. Formal Verification

In order to estimate the validity of the independent model of privacy release, in this section we propose a comparison with an abstraction of standard approaches in which information linking is irrevocable and privacy disclosure is incremental. Such a comparison is based on the evaluation of metrics that reveal how trading privacy for trust influences access to services and related costs.

For this purpose, we employ quantitative formal methods, thanks to which it is possible to estimate rigorously several properties of the system of interest, prior to implementation. The analysis is supported by the software tool PRISM [10],

[29]–[31], which is a model checker encompassing all the ingredients needed to model and verify our case study. More precisely, through PRISM it is possible to build automatically probabilistic models – like discrete-time Markov chains and Markov decision processes – from state-based formal specifications. The tool supports also modeling of stochastic multi-player games, in which nondeterministic choices are governed by distinct players, thus enabling explicitly the verification of different choice strategies. On the semantic models deriving from formal descriptions, quantitative properties expressed in probabilistic extensions of temporal logics can be verified through model checking techniques.

The comparison is conducted by assuming that the two models of privacy release are applied in the setting of a real-world cooperation system [9], in which users providing services, called *requestees*, and recipients of such services, called *requesters*, are involved in a cooperation process balancing trustworthiness of each participant with access to services and related costs. In the following, we briefly describe the original trust model and its relation with service remuneration [9]. Then, after introducing the modeling and verification assumptions, we discuss the analysis results.

A. Trust Model

Trust is a discrete metric with values ranging in the interval $[0, 50]$, such that $null = 0$, $low = 10$, $med = 25$, and $high = 40$. The trust T_{ij} of user i towards credential j (which can be, e.g., a crypto-id or an entity identity) is modeled abstractly as follows:

$$T_{ij} = \alpha \cdot trust_{ij} + (1 - \alpha) \cdot recs_{ij} \quad (1)$$

Parameter $\alpha \in [0, 1]$ is the risk factor balancing personal experience with recommendations by third parties. The trust metric $trust_{ij}$ is the result of previous direct interactions of i with j. Initially, $trust_{ij}$ is set to the dispositional trust of i, denoted by dt_i. After each positive interaction, $trust_{ij}$ is incremented by a factor v. Parameter $recs_{ij}$ is the average of the trust metrics towards j recommended to i by other users. For each service type, the service trust threshold *st* represents the minimum trust required to negotiate the service.

B. Service Cost Model

The joint combination of trust and remuneration is implemented by making the service cost function dependent on the trust T of the requestee towards the requester credential. The other main parameters are: C_{min}, which is the minimum cost asked by the requestee regardless of trust, C_{max}, which is the maximum cost asked to serve untrusted requests, and the threshold values T' and T'', such that $T'' < T'$.

The cost function proposed in [9] expresses linear dependence between trust and cost:

$$C(T) = \begin{cases} C_{min} + \frac{C_{max} - C_{min}}{T'} \cdot (T' - T) & \text{if } T < T' \\ C_{min} & \text{otherwise} \end{cases} \quad (2)$$

In order to examine thoroughly the trust/cost tradeoff, we consider two more functions approximating the linearity of the relation between trust and cost. In particular, a simple one-step function is as follows:

$$C(T) = \begin{cases} C_{max} & \text{if } T < T' \\ C_{min} & \text{otherwise} \end{cases} \quad (3)$$

while a possible two-steps function is as follows:

$$C(T) = \begin{cases} C_{max} & \text{if } T < T'' \\ C_{max}/2 & \text{if } T'' \leq T < T' \\ C_{min} & \text{otherwise} \end{cases} \quad (4)$$

C. Modeling Assumptions

Our objective is to compare the model of incremental release of privacy (represented in the figures by the curves named inc) with the model of independent release of privacy (represented in the figures by the curves named ind). For the sake of uniformity, for both models we assume abstractly that privacy is released (through the pseudonyms mechanism [19] and through the chunk mechanism, respectively) as a percentage of the total amount of sensitive information that the user may disclose. Similarly, in every trust-based formula we consider percentages of the trust involved.

The experiments are conducted by model checking several configurations of the system against formulas expressed in quantitative extensions of Computation Tree Logic [10]. For instance, Figure 1 refers to one requester interacting with one requestee with the aim of obtaining 10 services that can be of three different types. The figure reports the results for the best strategy, if one exists, allowing the requester to get access to all the services requested by minimizing the total expected cost (reported on the vertical axis) depending on the amount of revealed sensitive information (reported on the horizontal axis). The choice of the amount of privacy to spend for each request is under the control of the requester. The choice of the service type is either governed by the requester, or it is probabilistic with uniform distribution (see the curves denoted by $prob$ in the figure). Requestee's parameters are $dt = med$ and $v = 5$, as we assume that each transaction induces a positive feedback. The three service types are characterized by $st_1 = null$ and (2), $st_2 = low$ and (3), $st_3 = med$ and (4), respectively. The service cost parameters are $C_{min} = 0$, $C_{max} = 10$, $T' = high$, and $T'' = med$.

In order to focus on the difference between the two privacy models whenever the choice of the service is under the control of the requester, we also propose a sensitivity analysis with respect to parameter dt, where we concentrate on the interval of privacy values in which the previous experiment emphasizes the gap between the two models, see Figure 2.

We complete the comparison with an experiment assuming one requester and two requestees, which are chosen nondeterministically by the requester. The number of issued requests is 10, while we consider only the first type of service. The analysis, reported in Figure 3, proposes the results obtained by changing the service cost function. Requestee's trust parameters are as follows: $dt = med$, $st = null$, $\alpha = 0.5$.

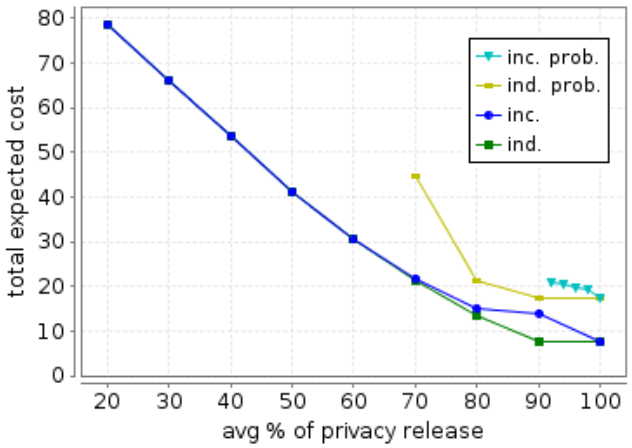
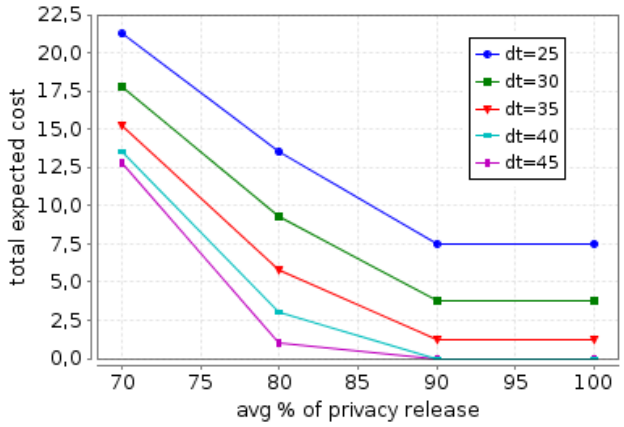
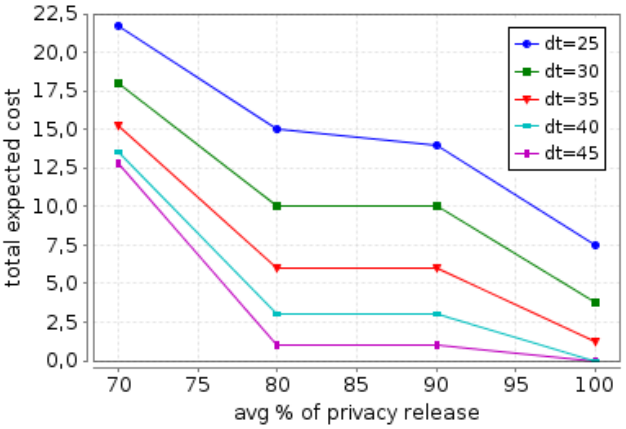


Figure 1. Trading cost for privacy.



(a) Independent privacy release.



(b) Incremental privacy release.

Figure 2. Trading cost for privacy by varying dispositional trust.

D. Evaluation

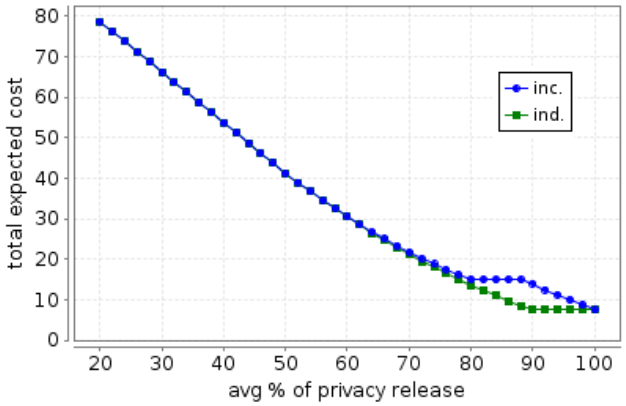
We now comment on the obtained results, by first considering Figure 1, which reveals two interesting behaviors.

Firstly, if the choice of the service is under the control of the requester, then the difference between the two models is significant only for values of the privacy release higher than 70%. In order to interpret this result, we checked the best requester's strategy, which consists of choosing always the service offering the best ratio trust/cost, i.e., the one using (2). Whenever trust is high enough to apply the minimum cost, then it turns out to be convenient to select also the other two service types. According to this strategy, if the privacy disclosure is below 70% it happens that trust does not reach the threshold T'. Therefore, as a consequence of (2), the relation between trust and cost is always linear and the two privacy models turn out to be equivalent from the economic standpoint. On the other hand, if the requester is highly trustworthy, then the cost to pay becomes constantly equal to the minimum cost, meaning that the requester could invest less privacy to obtain the same cost, thus revealing the advantages of the independent model. In practice, independently of the privacy model, it is economically convenient for the requester to disclose the information needed to obtain rapidly the best cost. Instead, for high levels of trust, it would be convenient for requester's privacy to reduce as much as possible the amount of disclosed information. Whenever identity of the requester is always fully disclosed, then the two models experience the same performance.

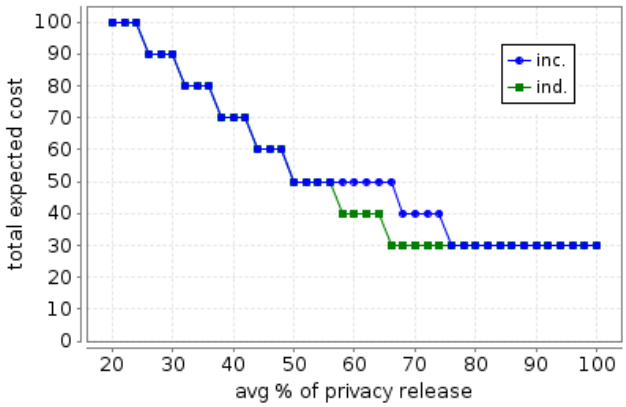
Secondly, if the choice of the service is probabilistic, thus modeling, e.g., a situation in which the requester may require every type of service independently of their cost, then it is not possible to satisfy all the requests if a minimum disclosure of privacy is not guaranteed. However, such a minimum value is considerably higher for the incremental model, in which case at least an average privacy release of 92% is needed. Hence, if the requester is somehow forced to require certain services, then the independent model performs better.

The analysis of Figure 2 confirms the results above also by varying the dispositional trust of the requestee, which is a parameter that does not affect the comparison between the two models. Different results are instead obtained by studying the role of the service cost function, as emphasized by the curves of Figure 3, which show that when step functions are used, the independent model is able to exploit better the intervals of trust in which the service cost is constant.

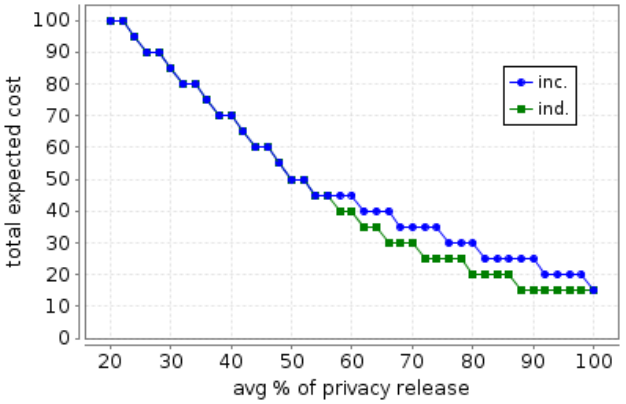
In the previous experiments, priority is given to service cost and to the average disclosure of privacy needed to optimize such a cost. However, if cost is not a fundamental issue, then the tradeoff of interest concerns trust and privacy. In order to analyze such a tradeoff, we reformulate the experiment of Figure 1 by focusing on the optimization of the average percentage of privacy release needed to obtain 10 services of a given type. The results are reported in Table I and refer to the second and third service types, for which the service trust threshold is *low* and *med*, respectively. Since to obtain such services the requester must be trusted by the requestee, we examine the tradeoff between such a trust and requester's privacy. For each of the two cases, the observed values show that through the independent model we obtain all the required services by disclosing much less privacy



(a) Cost Equation 2.



(b) Cost Equation 3.



(c) Cost Equation 4.

Figure 3. Trading cost for privacy by varying cost function.

than through the incremental model. The related difference is directly proportional to the trust threshold needed to obtain the services.

IV. Implementation Issues

The independent model of privacy release is based on the notion of virtual identity represented by means of the crypto-id, which we assume to be calculated using a cryptographic hash function, like SHA-3, over the public key of an asymmetric cryptography key pair generated by the user (see, e.g., [32] for a survey on cryptographic primitives). As a notation, we assume that (pk_u, sk_u) is the asymmetric crypto key pair associated with user u, such that pk_u is publicly available and $hash(pk_u)$ represents the related crypto-id.

TABLE I. Trading trust for privacy: avg % of privacy release.

service	inc.	ind.
type 2 (st_2 = *low*)	38%	28%
type 3 (st_3 = *med*)	92%	64%

Whenever issuing a service request, Alice chooses a bitmask that is applied to her crypto-id in order to extract the chunk according to the mechanism explained in Section II. Then, Alice sends to Bob a ciphertext (generated using Bob's public key) containing the chunk and a cryptographic proof for the request demonstrating that the chunk exposed is actually extracted from the crypto-id of the user issuing the request. In the following, we propose two solutions for the generation of such a proof that preserve anonymity of Alice's crypto-id.

In a centralized scenario, we assume that crypto-ids are stored in a non-public repository managed by a trusted, central authority (CA). In this case, the cryptographic proof may consist of a blind signature [33] obtained by Alice from the CA prior using the chunk, as explained in the following and shown in Figure 4. In the proposed protocol, (e_A, d_A) denotes an asymmetric crypto key pair generated by Alice to implement the blind signature.

Before issuing a service request associated with chunk C, Alice signs (with her private key sk_A) a request to the CA containing C and the encryption (using e_A) of the hash of a timestamp, $(C, (hash(t))_{e_A})_{sk_A}$. The request is accompanied by Alice's identity.

Upon reception of the validation request from Alice, the CA extracts C (using pk_A) and checks its conformity through the public key of Alice, by comparing C against the crypto-id $hash(pk_A)$. Then, if such a check is successful, the CA generates a blind signature including C and the encrypted hashed timestamp, $(C, (hash(t))_{e_A})_{sk_{CA}}$. Notice that the CA can neither guess the timestamp t, nor associate its hashed value with Alice.

When receiving such a ciphertext, Alice strips away her encryption using d_A, thus leaving her with the CA signature of C and of the hash of the timestamp, $(C, (hash(t)))_{sk_{CA}}$.

The request sent to Bob includes C, the timestamp, and the CA signature, $(C, t, (C, (hash(t)))_{sk_{CA}})$.

Hence, Bob can check the signature for validity, by comparing C and t against the content of the message signed by the CA, and then forward the obtained information to the CA, which verifies timestamp doublespending. The reason for using a timestamp is to avoid unauthorized users employing chunks signed by the CA, while the use of the blind signature ensures Alice anonymity. Indeed, notice that the knowledge of C does not allow neither Bob nor the CA to infer the originating crypto-id and, therefore, the identity of Alice. More

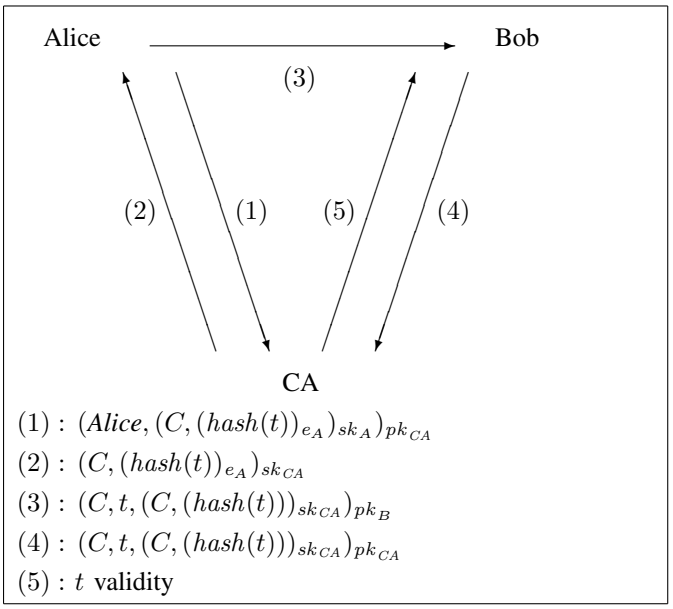


Figure 4. Cryptographic proof of chunk C through the CA.

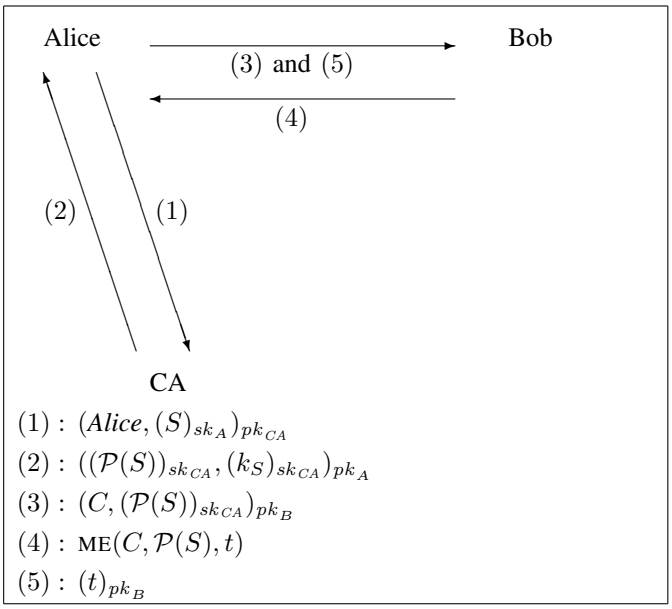


Figure 5. Cryptographic proof of chunk C in zero-knowledge.

sophisticated blind signature schemes can be used, e.g., to offer fairness [34], in order to revoke blindness in case of suspicious behaviors by some chunk and, therefore, isolate dishonest users.

Alternatively, zero-knowledge proofs can be applied on-the-fly whenever we consider a distributed scenario that does not involve communication with the CA during the transaction lifetime. For instance, zero-knowledge sets and set membership [35], [36] are proposed to decide the membership problem $x \in S$ by preserving as much privacy as possible about S (or x). In particular, a zero-knowledge membership proof works as follows. Let $\mathcal{P}(S)$ be a privacy-preserving token of a set S (e.g., a certified commitment by the CA on the set S that does not reveal any information about its constituting elements) and x an element belonging to S. Whenever the verifier knows the pair $(\mathcal{P}(S), x)$, the prover can convince the verifier in zero-knowledge that $x \in S$ without leaking anything about S to the verifier. Membership encryption [37] is a cryptographic technique extending membership proof in which:

- $\mathcal{P}(S)$ is generated from S and a secret key k_S;
- the encryption algorithm, called ME, takes as input x, $\mathcal{P}(S)$, and the message m to encrypt;
- the decryption algorithm, called MD, requires the pair (S, k_S) and holding the membership $x \in S$ to return successfully m.

In our setting, x is represented by the chunk C used by Alice, while S is given by the set of chunks committed by the CA whenever Alice registers her crypto-id in the CA repository. Hence, Bob plays the role of verifier whenever Alice exposes chunk C and the certified token $\mathcal{P}(S)$. The handshake works as illustrated in Figure 5.

Initially, Alice signs a request to the CA including the list S of chunks she intends to use in future interactions (such a request can be renewed if Alice requires more chunks).

The CA verifies whether S is correct with respect to Alice's crypto-id, i.e., each of its elements matches the hash of the public key of Alice. If this is the case, the CA generates the privacy-preserving token $\mathcal{P}(S)$ and the secret key k_S, and then signs such a pair for Alice.

Afterwards, when issuing a request to Bob, Alice sends the chosen chunk C, which is expected to belong to S, and the certified token $\mathcal{P}(S)$.

Bob calculates $\text{ME}(C, \mathcal{P}(S), t)$, where t is a timestamp chosen by Bob, and then asks Alice to extract successfully t to prove that $C \in S$.

Finally, Alice computes $\text{MD}(\text{ME}(C, \mathcal{P}(S), t), S, k_S)$, which is equal to t if and only if $C \in S$. It is worth noticing that the membership proof from membership encryption is non-transferable, i.e., Bob cannot convince any third party that $C \in S$, thus ensuring the privacy of $\mathcal{P}(S)$.

Once Bob accepts a request accompanied by chunk C, he must estimate trustworthiness towards C in order to negotiate the service parameters. In the following, we propose a centralized reputation based approach and a distributed trust based approach. To this aim, we assume to deal with a numeric, totally ordered domain $\mathcal{T}$ for trust and reputation values and that the evaluation feedback at the end of every transaction is reported as a positive/negative variation.

A. Design of a Centralized Reputation System

The key feature of the proposed approach to privacy management is that any transaction is associated with portions, called chunks, of the crypto-id representing the virtual identity. The same chunk can be shared by different users, who decide for each transaction the chunk size and whether to combine

together chunks previously used. Hence, the relation among chunks and related crypto-ids must be managed carefully in order to estimate correctly the reputation of users.

For this purpose, the centralized reputation system we propose is managed by the CA, which is in charge of two main tasks:

1) management of the reputation of each crypto-id on the basis of the feedback reported about the chunks that match the crypto-id;
2) calculation of the reputation of the chunk spent in a transaction on the basis of the reputations of the crypto-ids matching the chunk.

Since in a limiting scenario a chunk could be a single bit, based on such a granularity we assume that reputation is managed at the bit level.

As far as the first task of the CA is concerned, whenever at the end of a transaction a user transmits the feedback concerning a chunk C, the CA is not able to infer from which crypto-id C is actually originated. Hence, the CA distributes the result v of the user evaluation among the bits of C for every crypto-id matching C. More precisely, the bit reputation variation is $\delta \cdot v$, where δ is a discounting factor in $[0, 1]$ proportional to the size of the chunk. On one hand, the role of δ is to strengthen the relation between the amount of sensitive information exposed by the user in a transaction and the trustworthiness towards such a user. On the other hand, δ mitigates the effect of the use of small chunks, as they are shared by a larger number of users and, therefore, they represent very roughly the users employing them.

Example 2. *Consider four users with the following crypto-ids:*

$$K_1 : 10010100 \quad K_2 : 00010010$$
$$K_3 : 01110111 \quad K_4 : 11011011$$

and an initial situation in which the vector of bit's reputation is $rep_{K_i} = 00000000$*, for* $1 \leq i \leq 4$*. If user* 1 *employs bitmask* 01110000 *for a transaction evaluated positively with* $v = 1$ *(and* $\delta = 1$*), then, the update performed by the CA is* $rep_{K_1} = 01110000$ *and* $rep_{K_2} = 01110000$*, because the chunk used is shared by users* 1 *and* 2*.*

Then, if user 3 *uses bitmask* 00011100 *and the feedback is as above, we obtain the reputation changes* $rep_{K_1} = 01121100$ *and* $rep_{K_3} = 00011100$*.*

Finally, if user 4 *uses bitmask* 00000111*, then any feedback is applied to (the first three bits of)* K_4 *only.*

As shown by the example, the choice of the chunk does not ensure perfect privacy of the user with respect to the CA. Similarly as discussed in Section II, the probability of an unequivocal identification of the crypto-id by the CA depends on the size of the community and of the chunk.

As far as the second task of the CA is concerned, the calculation of the reputation of a chunk deals with the same issues surveyed above. Whenever a user forwards to the CA a chunk C in order to know the related reputation, the CA could not be able to infer the identity of the originating crypto-id. Thus, the reputation of C results from a combination (through the arithmetic mean) of the reputations of such a chunk within every crypto-id K matching C.

Let $rep_K(C)$ be the reputation of chunk C within the crypto-id K, which is calculated by summing up the reputations of the bits of K forming C. By default, $rep_K(C) = 0$ if K does not match C. Moreover, let $\mathcal{M}(C)$ denote the number of crypto-ids matching C. Then, the reputation of chunk C is:

$$\frac{1}{\mathcal{M}(C)} \cdot \sum_K rep_K(C) \tag{5}$$

where the summation is over all the crypto-ids K registered in the CA repository.

Example 3. *With reference to the previous example, consider a new transaction in which user* 1 *employs the bitmask* 01110000*. The resulting chunk is shared by users* 1 *and* 2*. Hence, by using (5), its reputation is:*

$$\tfrac{1}{2} \cdot ((1 + 1 + 2) + (1 + 1 + 1)) = 3.5$$

B. Design of a Distributed Trust System

Handling trust towards users by tracing the usage of (possibly shared) chunks is a hard task in the absence of a centralized reputation system. To deal with this problem, in order to estimate user's trustworthiness we define a local trust structure that allows any user offering a service to associate a trust value with every chunk received to negotiate the service. In particular, the proposed approach does not rely on the knowledge of the list of crypto-ids.

Let $\mathcal{C}$ be the set of chunks with which the user has interacted in completed transactions. The local trust structure is based on the definition of a partially ordered set (*poset*, for short) $(\mathcal{C}, \leq)$ over set $\mathcal{C}$ with respect to a partial order $\leq$. We recall that to define a poset, the binary relation $\leq$ must be reflexive, antisymmetric, and transitive for the elements of $\mathcal{C}$. In the rest of the section, we call $\leq$ refinement operator, which is defined as follows.

Definition 1 (Chunk refinement)**.** *Let* n *be the crypto-id size. Given chunks* $C_B, C_{B'}$*, we say that* $C_{B'}$ *refines* C_B*, denoted* $C_B \leq C_{B'}$*, if and only if:*

- *for all* $1 \leq i \leq n$*:* $B[i] \leq B'[i]$*;*
- *for all* $1 \leq i \leq n$*: if* $B[i] = 1$ *then* $C_B[i] = C_{B'}[i]$*.*

Notice that if $C_B \leq C_{B'}$ then B is a submask of B' and the information exposed by $C_{B'}$ includes that revealed by C_B. The intuition is that if two chunks are related through $\leq$ then they could be originated from the same crypto-id.

As we will see, maintaining the poset structure provides the means to approximate the trust towards any crypto-id by employing the trust related to the potential constituting chunks. Each element of the poset $(\mathcal{C}, \leq)$ is labeled by a value of the trust domain $\mathcal{T}$. Such a value represents the trust of the user towards the related chunk resulting from interactions associated with such a chunk. Formally, we denote such an extended structure with $(\mathcal{C}, \leq, t)$, where $t : \mathcal{C} \to \mathcal{T}$ defines the mapping from chunks to trust values. Initially, for every unknown chunk C with which the user interacts for the first time, we assume $t(C)$ to be equal to the dispositional trust dt of the user, which represents the attitude to cooperate with unknown users.

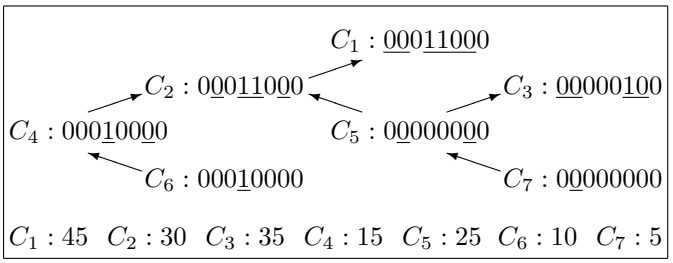


Figure 6. Example of a local trust structure.

Example 4. *Figure 6, which in the following we use as running example, shows the graphical representation of a poset, where, e.g., $C_6 \leq C_4 \leq C_2 \leq C_1$, as well as $C_7 \leq C_5 \leq C_3$, while, e.g., C_6 and C_3 are not related with each other. Moreover, the figure reports also the trust associated with each known chunk at a given instant of time, by assuming the trust domain* $[0, 50]$.

To emphasize the nature of the independent model of privacy release, notice that even if Alice invested chunk C_1 in a past interaction with Bob, whose reference trust structure is that depicted in Figure 6, then in the current transaction she may use chunk C_2 only, while Bob cannot infer the link between the user of the past interaction associated with C_1 and the current one. As a side effect, notice also that all the users with a crypto-id matching C_2 actually benefit from the trust (or pay the mistrust) associated with C_2.

The obfuscation mechanism illustrated in the example above respects the requirements of the independent model of privacy release discussed in Section II.

Similarly as done in the previous section, we now illustrate how to manage the trust $t(C)$ towards the chunk C on the basis of the feedback v following any transaction associated with C. In particular, the trust variation applied to $t(C)$ is simply $\delta \cdot v$, where δ is the discounting factor discussed in the previous section.

Example 5. *As a consequence of a positive transaction conducted through chunk C_2 and resulting in a trust variation equal to, e.g., $+5$, we would obtain $t(C_2) = 32.5$ if $\delta = 0.5$, and $t(C_2) = 35$ if $\delta = 1$. Notice that in the former case the discounting factor represents the ratio between the size of the chunk used and the size of the originating crypto-id, thus emphasizing that trust is proportional to the amount of information disclosure.*

Once $t(C)$ has been updated by applying the variation $\delta \cdot v$, it is worth deciding whether and how the feedback related to chunk C has to be propagated to other elements of the trust structure $(\mathcal{C}, \leq, t)$. First of all, propagation would result in ambiguity if applied to chunks of the poset that cannot be related through $\leq$, because unrelated chunks cannot be brought back to the same crypto-id. Therefore, the remaining cases refer to the chunks that refine (or are refined by) C.

Depending on the feedback, which can be either positive or negative, the potential application of a discounting factor, and the propagation direction (towards finer or coarser chunks, or else both), every possible combination gives rise to a different propagation policy. Tuning these parameters is a task of the user depending on her/his attitude to cooperation. In the following, we describe a policy balancing accuracy of the trust estimations with robustness against malicious behaviors.

On one hand, negative trust variations are not propagated to elements that refine C, because an interaction disclosing a small amount of sensitive information should not compromise the trust level of chunks that expose more information. The objective of this rule is to contrast potential attacks by users preserving their identity and aiming at penalizing the trust of small chunks shared by a large number of users. On the other hand, in order to overcome the problem of trust underestimation and to fully exploit the flexibility of the independent model of privacy release, positive trust variations are propagated to chunks refining C, while positive/negative trust variations are propagated to every chunk in the poset that is refined by C. Another objective of this rule is to favor, in terms of trust, the disclosure of information. In order to keep under control the propagation mechanism, the trust variation for any chunk C' inherited by the feedback related to chunk C is further discounted by a factor δ' proportional to the difference between the size of C and the size of C'. In practice, the larger the difference between C and C' is, the slighter the impact of the trust variation of C upon C'.

Example 6. *Consider chunk C_2 and the positive transaction of the previous example determining $t(C_2) = 32.5$ (i.e., $\delta \cdot v = 2.5$). Then, by virtue of the propagation policy discussed above we have, e.g., $t(C_4) = 16.25$ and $t(C_6) = 10.625$. Chunk C_5 (resp. C_7) gains the same variation applied to C_4 (resp., C_6). On the other hand, C_1 inherits a discounted trust gain equal to $2.5 \cdot \frac{2}{3}$, because C_1 refines C_2 and the trust variation is positive, while C_3 does not inherit any trust gain, because C_2 and C_3 are not related with each other.*

The local trust structure continuously evolves not only by virtue of the updates discussed above, but also as a consequence of the treatment of new chunks. Associating a new chunk C that is added to the poset with the dispositional trust of the user is a policy that does not take into account the knowledge of the trust structure $(\mathcal{C}, \leq, t)$, which can be employed to infer some trust information about C.

Based on the same intuition behind feedback propagation, the trust values associated with known chunks that are in some relation with C can be combined to set up the initial value of $t(C)$. In fact, C can be interpreted as an approximation of such chunks. As in the case of the propagation policy, we can envision several different rules, among which we advocate the following one: $t(C)$ is assigned the arithmetic mean of the trust values associated with chunks that refine C, while those refined by C are ignored. In fact, the accuracy of the trust estimations is directly proportional to the size of the chunks. Therefore, estimating $t(C)$ based on small chunks refined by C would lead to a rough approximation of the trust towards the users employing C. Moreover, the chunks refining C that are considered must be pairwise unrelated by $\leq$ in order to avoid redundancy when counting the related trust values.

Definition 2 (Chunk coverage)**.** *Let $(\mathcal{C}, \leq, t)$ be a trust structure and $C \notin \mathcal{C}$ a new chunk that must be added to the poset. A coverage for C is a set $\mathcal{K} = \{C_1, \ldots C_m\} \subseteq \mathcal{C}$ such that:*

- *$C_i \not\leq C_j$ for all $1 \leq i, j \leq m$;*
- *$C \leq C_i$ for all $1 \leq i \leq m$.*

The initial value of $t(C)$ induced by the coverage $\mathcal{K}$ is:

$$t_{\mathcal{K}}(C) = \frac{1}{m} \cdot \sum_{i=1}^{m} t(C_i).$$

It is worth noticing that the poset may enable several different coverages for a chunk C. If $\mathcal{K}_1, \ldots, \mathcal{K}_p$ are the possible coverages for C in the trust structure $(\mathcal{C}, \leq, t)$, then whenever C is added to $\mathcal{C}$ we set:

$$t(C) = f\{\!|\, t_{\mathcal{K}_i}(C) \mid 1 \leq i \leq p \,|\!\}$$

where f is an associative and commutative arithmetical function (like, e.g., min, max, and avg) applied to the multiset of initial trust values induced by the different coverages.

Example 7. *Consider the trust structure of Figure 6. A coverage for chunk C_8 : 00000000 is the set $\mathcal{K} = \{C_4, C_5\}$, which induces the initial trust value $t_{\mathcal{K}}(C_8) = 20$. Other candidates are $\{C_2, C_3\}$, $\{C_3, C_4\}$, and $\{C_1\}$. Therefore, the initial trust resulting from the application of function avg is 30.625, while we obtain 45 for function max and 20 for function min.*

In general, from the effectiveness standpoint, the trust structure $(\mathcal{C}, \leq, t)$ is used to manage locally information (about chunk's trust) allowing the user to approximate the trust towards other users, without any knowledge about their crypto-ids and actual behaviors. As far as efficiency issues are concerned, in order to circumvent the problem of dealing with a huge trust structure, it is possible to constrain a priori the number of different chunks that can be chosen by every user.

C. Evaluation

The chunk based identity sharing mechanism of the independent model of privacy release has several impacts upon the functionalities of the reputation and trust systems. As a consequence of chunk sharing, the crypto-ids matching the same chunk actually benefit from the reputation (or pay the mistrust) associated with such a chunk. This aspect is crucial for the requirements of the independent model of privacy release and can be viewed as an incentive to take honest decisions, because a high number of trustworthy chunks contribute to increase the probability of obtaining services at a reasonable cost by preserving the desired level of privacy. Hence, all the users sharing trustworthy chunks benefit from this virtuous circle.

Obviously, it is beneficial for a chunk C if all users controlling it are trustworthy. However, if at least one of them is very untrustworthy and carries out at least one illegal action linked to C, then the chunk may rapidly become untrustworthy and useless for all other users. In addition, if C becomes untrustworthy then it may also impact the trustworthiness of any chunk C' such that C and C' match the crypto-id of the same user. These side effects are mitigated implicitly by using mixed cooperation strategies based on trust and cost [14], [16], [17] and explicitly by applying the discounting factors discussed in the previous sections. An effective but severe solution consists of resorting to a CA capable of revoking blindness in case of suspicious behaviors by some chunk, in order to isolate dishonest users and repair the reputation of the chunk involved.

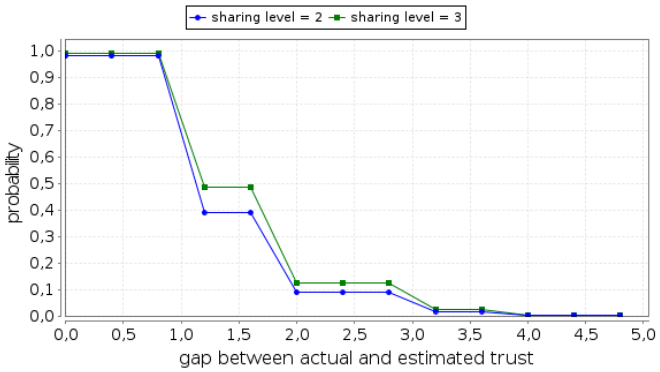


Figure 7. Approximating user trust through chunk trust.

The choice of the chunk size represents another important aspect. In fact, a tradeoff exists between chunk size, privacy, and trust/reputation. The user privileging privacy employs chunks of small size. With high probability, small chunks cannot be used to negotiate favorable service conditions and provide also a rough approximation of the real trustworthiness of the user, as they are shared by a high number of users influencing their usage. On the other hand, the user privileging accuracy of trustworthiness employs chunks of large size, thus sacrificing more privacy as the probability of identification becomes higher.

The discussion above emphasizes that the use of chunks (and of trust information based on them) implies an approximation of the estimation of the trust towards users. In order to quantify the approximation level, an experiment has been conducted by employing the formal framework illustrated in Section II. By assuming a scenario with 4 users, each one having 2 chunks to issue 25 total requests to a service provider, we have evaluated the difference between the *estimated* trustworthiness of each user (as resulting from the combination of the trust of each chunk matching the user crypto-id) and the *actual* trustworthiness of each user (that derives by tracing the actual behavior of the user). For each service request, uniform probability distributions have been used to govern the choice of: the user negotiating the transaction, the chunk exposed, and the feedback reported about the user behavior (no discounting factor is applied). Moreover, we recall that the trust domain is the interval $[0, 50]$.

For this scenario, the curves of Figure 7 evaluate the probability with which the (absolute value of the) difference between actual and estimated trust is higher than the values reported in the horizontal axis. Each curve refers to a different sharing level, which expresses the minimum number of users sharing every chunk. For instance, we observe that the probability that

the trust gap is greater than 2 for a sharing level equal to 2 (resp., 3) is less than 10% (resp., 12%) and rapidly converges to zero.

Finally, the combination of the trust and reputation systems surveyed above can be easily achieved by merging the resulting metrics through the following formula:

$$\alpha \cdot trust(C) + (1 - \alpha) \cdot rep(C) \quad (6)$$

where C is the chunk under evaluation, α is the risk factor, function $trust$ returns the trust resulting from the distributed trust system, and function rep returns either the reputation provided by the CA if a centralized reputation system is available, or a combination (through the arithmetic mean) of the trust values possibly recommended by neighbors. Moreover, we emphasize that the presentation of the proposed design models abstracts away from the specific trust and reputation metrics that are adopted. Indeed, basically, our method may be integrated with any computational notion of trust and with any recommendation mechanism used in classical trust/reputation systems, see, e.g., [38]–[40].

V. Conclusion

The attitude to cooperation is strongly affected by the tradeoff existing among privacy and trustworthiness of the involved parties and cost of the exchanged services. The proposed model of privacy release offers a high level of flexibility in the management of such a tradeoff. In particular, by virtue of a mechanism based on the splitting of crypto-ids, it is possible to manage the disclosure of sensitive information in a less restrictive way with respect to classical models.

To summarize the results obtained from the formal verification, we observe that the major freedom degree of the independent model ensures better performance with respect to the incremental model. This is always true if the main objective is trading privacy for trust. If services must be paid and cost depends on trust, then the adopted cost function affects the tradeoff among privacy, trust, and cost, by revealing the advantages of the independent model in the intervals of trust values in which cost is constant.

From the implementation viewpoint, it has been shown that the novel model can be effectively applied both in centralized reputation systems and in distributed trust systems. The empirical analysis of the peculiarities of each solution, like the bottleneck problem induced by the CA or the efficiency and accuracy ensured by the local trust structure, represents work in progress.

We conclude by observing that a successful deployment of the proposed approach is strictly related to the choice of the trust policies and configuration parameters, which are currently subject to sensitive analysis through formal verification. Solutions to manage the dynamic variability at run time of these parameters are left as future work. Similarly, the approximation induced by the analysis of chunk trustworthiness whenever estimating the actual behavior of users shall be verified in a real-world scenario characterized by a sufficiently large population.

References

[1] A. Aldini, "Saving privacy in trust-based user-centric distributed systems," in 8th Int. Conf. on Emerging Security Information, Systems and Technologies (SECURWARE2014). IARIA, 2014, pp. 76–81.

[2] A. Aldini, A. Bogliolo, C. Ballester, and J.-M. Seigneur, "On the tradeoff among trust, privacy, and cost in incentive-based networks," in 8th IFIP WG 11.11 Int. Conf. on Trust Management, ser. IFIP AICT, J. Zhou et al., Eds., vol. 430. Springer, 2014, pp. 205–212.

[3] A. Aldini and A. Bogliolo, Eds., User-Centric Networking – Future Perspectives, ser. Lecture Notes in Social Networks. Springer, 2014.

[4] A. Jøsang, "Trust and reputation systems," in Foundations of Security Analysis and Design IV (FOSAD'07), ser. LNCS, A. Aldini and R. Gorrieri, Eds. Springer, 2007, vol. 4677, pp. 209–245.

[5] J. Sabater and C. Sierra, "Review on computational trust and reputation models," Artificial Intelligence Review, vol. 24, 2005, pp. 33–60.

[6] E. Chang, F. Hussain, and T. Dillon, Trust and Reputation for Service-Oriented Environments: Technologies For Building Business Intelligence And Consumer Confidence. Wiley, 2005.

[7] S. Greengard, "Social games, virtual goods," Communications of the ACM, vol. 54, no. 4, 2011, pp. 19–22.

[8] S. Taddei and B. Contena, "Privacy, trust and control: Which relationships with online self-disclosure?" Computers in Human Behavior, vol. 29, no. 3, 2013, pp. 821–826.

[9] A. Bogliolo, P. Polidori, A. Aldini, W. Moreira, P. Mendes, M. Yildiz, C. Ballester, and J.-M. Seigneur, "Virtual currency and reputation-based cooperation incentives in user-centric networks," in 8th Int. Wireless Communications and Mobile Computing Conf. (IWCMC'12). IEEE, 2012, pp. 895–900.

[10] V. Forejt, M. Kwiatkowska, G. Norman, and D. Parker, "Automated verification techniques for probabilistic systems," in Formal Methods for Eternal Networked Software Systems, ser. LNCS, M. Bernardo and V. Issarny, Eds. Springer, 2011, vol. 6659, pp. 53–113.

[11] L. Caviglione, M. Coccoli, and A. Merlo, Eds., Social Network Engineering for Secure Web Data and Services. IGI Global, 2013.

[12] Y. Zhang, L. Lin, and J. Huai, "Balancing trust and incentive in peer-to-peer collaborative system," Journal of Network Security, vol. 5, 2007, pp. 73–81.

[13] M. Yildiz, M.-A. Khan, F. Sivrikaya, and S. Albayrak, "Cooperation incentives based load balancing in UCN: a probabilistic approach," in Global Communications Conf. (GLOBECOM'12). IEEE, 2012, pp. 2746–2752.

[14] Z. Li and H. Shen, "Game-theoretic analysis of cooperation incentives strategies in mobile ad hoc networks," Transactions on Mobile Computing, vol. 11, no. 8, 2012, pp. 1287–1303.

[15] A. Aldini and A. Bogliolo, "Model checking of trust-based user-centric cooperative networks," in 4th Int. Conf. on Advances in Future Internet (AFIN2012). IARIA, 2012, pp. 32–41.

[16] A. Aldini, "Formal approach to design and automatic verification of cooperation-based networks," Journal On Advances in Internet Technology, vol. 6, 2013, pp. 42–56.

[17] M. Kwiatkowska, D. Parker, and A. Simaitis, "Strategic analysis of trust models for user-centric networks," in Int. Workshop on Strategic Reasoning (SR'13), vol. 112. EPTCS, 2013, pp. 53–60.

[18] A. Aldini and A. Bogliolo, "Modeling and verification of cooperation incentive mechanisms in user-centric wireless communications," in Security, Privacy, Trust, and Resource Management in Mobile and Wireless Communications, D. Rawat, B. Bista, and G. Yan, Eds. IGI Global, 2014, pp. 432–461.

[19] J.-M. Seigneur and C.-D. Jensen, "Trading privacy for trust," in 2nd Int. Conf. on Trust Management (iTrust'04), ser. LNCS, vol. 2995. Springer, 2004, pp. 93–107.

[20] V. Sassone, S. Hamadou, and M. Yang, "Trust in anonymity networks," in Conf. on Concurrency Theory (CONCUR'10), ser. LNCS, vol. 6269. Springer, 2010, pp. 48–70.

[21] M. Raya, R. Shokri, and J.-P. Hubaux, "On the tradeoff between trust and privacy in wireless ad hoc networks," in 3rd ACM Conf. on Wireless Network Security (WiSec'10), 2010, pp. 75–80.

[22] L. Lilien and B. Bhargava, "Privacy and trust in online interactions," in Online Consumer Protection: Theories of Human Relativism. IGI Global, 2009, pp. 85–122.

[23] W. Wagealla, M. Carbone, C. English, S. Terzis, and P. Nixon, "A formal model of trust lifecycle management," in Workshop on Formal Aspects of Security and Trust (FAST'03), 2003.

[24] S. Köpsell and S. Steinbrecher, "Modeling unlinkability," in 3rd Workshop on Privacy Enhancing Technologies, ser. LNCS, vol. 2760. Springer, 2003, pp. 32–47.

[25] I. Goldberg, "A pseudonymous communications infrastructure for the internet," Ph.D. dissertation, University of California at Berkeley, 2000.

[26] A. Kobsa and J. Schreck, "Privacy through pseudonymity in user-adaptive systems," ACM Transactions on Internet Technology, vol. 3, no. 2, 2003, pp. 149–183.

[27] "Identity Mixer," accessed: 2015-05-15. [Online]. Available: http://www.zurich.ibm.com/idemix/

[28] C. Paquin and G. Zaverucha, "U-Prove cryptographic specification v1.1 (revision 3)," 2013, accessed: 2015-05-15. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=166969

[29] T. Chen, V. Forejt, M. Kwiatkowska, D. Parker, and A. Simaitis, "Prism-games: a model checker for stochastic multi-player games," in 19th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'13), ser. LNCS, vol. 7795. Springer, 2013, pp. 185–191.

[30] ——, "Automatic verification of competitive stochastic systems," Formal Methods in System Design, vol. 43, no. 1, 2013, pp. 61–92.

[31] M. Kwiatkowska, G. Norman, and D. Parker, "Prism 4.0: verification of probabilistic real-time systems," in 23rd Int. Conf. on Computer Aided Verification (CAV'11), ser. LNCS, vol. 6806. Springer, 2011, pp. 585–591.

[32] J. Katz and Y. Lindell, Introduction to Modern Cryptography – 2nd Edition. CRC Press, 2014.

[33] D. Chaum, A. Fiat, and M. Naor, "Untraceable electronic cash," in Advances in Cryptology (CRYPTO'88), ser. LNCS, vol. 403. Springer, 1990, pp. 319–327.

[34] M. Stadler, J.-M. Piveteau, and J. Camenisch, "Fair blind signatures," in Eurocrypt'95, ser. LNCS, vol. 921. Springer, 1995, pp. 209–219.

[35] S. Micali, M. Rabin, and J. Kilian, "Zero-knowledge sets," in 44th Symposium on Foundations of Computer Science (FOCS2003). IEEE, 2003, pp. 80–91.

[36] J. Camenisch, R. Chaabouni, and A. Shelat, "Efficient protocols for set membership and range proofs," in 14th Int. Conf. on the Theory and Application of Cryptology and Information Security (ASIACRYPT'08), ser. LNCS, vol. 5350. Springer, 2008, pp. 234–252.

[37] F. Guo, Y. Mu, W. Susilo, and V. Varadharajan, "Membership encryption and its applications," in 18th Australasian Conf. on Information Security and Privacy (ACISP2013), ser. LNCS, C. Boyd and L. Simpson, Eds., vol. 7959. Springer, 2013, pp. 219–234.

[38] S.-D. Kamvar, M.-T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in p2p networks," in 12th Conf. on World Wide Web (WWW'03). ACM, 2003, pp. 640–651.

[39] R. Zhou and K. Hwang, "Powertrust: a robust and scalable reputation system for trusted peer-to-peer computing," Transactions on Parallel and Distributed Systems, vol. 18, no. 4, 2007, pp. 460–473.

[40] M. Gupta, P. Judge, and M. Ammar, "A reputation system for peer-to-peer networks," in 13th Int. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'03). ACM, 2003, pp. 144–152.