Learning Whom to Trust in a Privacy-Friendly Way

Sebastian Ries, Marc Fischlin, Leonardo A. Martucci, Max Mühlhäuser
Center for Advanced Security Research Darmstadt (CASED)
Telecooperation Group (TK) — Technische Universität Darmstadt
{firstname.lastname}@cased.de

***Abstract*—The topics of trust and privacy are more relevant to users of online communities than ever before. Trust models provide excellent means for supporting users in their decision making process. However, those models require an exchange of information between users, which can pose a threat to the users' privacy. In this paper, we present a novel approach for a privacy preserving computation of trust. Besides preserving the privacy of the recommenders by exchanging and aggregating recommendations under encryption, the proposed approach is the first that enables the trusting entities to learn about the trustworthiness of their recommenders at the same time. This is achieved by linking the minimum amount of information that is required for the learning process to the actual recommendation and by using zero-knowledge proofs for assuring the correctness of this additional information.**

I. Introduction

Trust and reputation systems support users' in making decisions whether and with whom to interact in online environments: "Reputation is a summary of one's past actions within the context of a specific community, presented in a manner that can help other community members make decisions with respect to whether and how to relate to that individual." [1] Those systems became more and more important and ubiquitous in eCommerce, virtual communities, virtual organizations, and in Internet of Services. For example, a recent study shows that eBay's data center, which manages all transactions and revenue from eBay.com and PayPal.com, processed a total of $60 billion in 2009, i.e. around $2,000 per second.[1]

On the other hand, privacy is of fundamental importance in computer-based environments. Privacy can be defined in terms of a person's right to determine when, how and to what extent information about him or her is communicated to the others [2]. The principle of necessity of data collection and processing determines that the collection and processing of personal data should only be allowed if it is necessary for the tasks falling within the responsibility of the data processing agency. Hence, personal information should not be collected or used for identification purposes when not absolutely necessary. The best strategy to enforce such requirement is the avoidance or minimization of personal data [3], [4].

[1] `http://www.informationweek.com/blog/global-cio/229202763`

There is a conflict between mechanisms for trust establishment and for privacy protection. While trust mechanisms require information exchange and long-term identifiers, privacy enhancing technologies aim to limit usage of personal data using properties like anonymity or short-term (transaction) pseudonyms. This inherent conflict is the result of opposing interests derived from the aforementioned principle of necessity of data collection and processing, and the principle of data minimization [5].

The exchange of recommendations poses a special threat to privacy. In particular, if recipients can link recommendations across multiple service contexts, then they can learn about the past interactions and preferences of the recommenders. Hence, the problem between trust establishment and privacy protection.

In this paper, we address this problem by proposing a novel approach for exchanging information that is based on a pre-agreed set of recommenders and homomorphic encryption. We extend the current state of the art with means to compute trust while preserving the users' privacy. Our proposal updates trust in recommenders based on the accuracy of their recommendations and without offering a full-knowledge of the recommendation. This aspect has not been addressed before to the best of our knowledge. Our solution is based on the idea to extend a recommendation with an additional piece of information. This additional information states the "tendency" of the recommendation and also proves the link between them using zero knowledge proofs.

All in all, the paper presents a privacy-preserving trust mechanisms that supports the following:

- Learning about the trustworthiness of one's recommenders – without knowing the details of the recommendations.
- Allowing members of the set of recommenders to say "I don't know" without skewing the calculated trust value or the need to adjust the set of recommenders.
- Distribution of negative evidence. Negative evidences are fundamental in reputation systems. They allow reputation values to fluctuate according to the user's behavior. Without negative evidences, malicious users that build a good reputation could misbehave freely, with no damaging effect on their reputation.

The remainder of the paper is structured as follows. Section II summarizes the related work. An approach

for computing trust values is presented in Section III and the objectives of the paper are then formalized in Section IV. Section V describes our protocol for a privacy-preserving computation of trust. The proofs and properties are presented in Section VI and the cryptographic tools are detailed in Section VII.Section VIII concludes the paper.

II. Related Work

In this section we introduce the relevant related work. This section is divided into four parts. First, we recapitulate evidence based trust models. Second, we discuss how privacy is treated according to the trust model used. Third, we organize possible solutions according to their underlying theoretical fundamentals. Finally, we briefly discuss the cryptographic tools that we used in this paper and mention other works that are relevant to the field.

A. Evidence based trust models

Evidence-based trust and reputation models try to estimate the trustworthiness (or reputation) of a service provider based on the evidence that reflects the outcome of previous interactions of a customer with the service. One of the most prominent examples of such an approach is the feedback score that supports buyers when selecting sellers on eBay (`www.eBay.com`). Furthermore, there has been a number of proposals of trust and reputation models in research, e.g., centralized models [6]–[8], and distributed models [9]–[13].

Considering the representation and computation of trust, Bayesian trust models [7], [9], [10], [12], [14] provide a couple of advantages compared to ad hoc developed approaches: *(i)* they provide sound mathematical basis and simple mechanism to update trust values whenever new evidence is available, and *(ii)* they allow to consider the amount of information at trust value is based on, especially, they allow to explicitly express that one does not have any information,[2] *(iii)* they allow to interpret trust as a subjective probability which fits the definition of trust provided in [15].

B. Dealing with privacy in trust models

In the following we like to discuss the privacy issues of trust models considering the location where trust values are stored and computed:

- **Centralized Models:** In trust models like eBay or the Bayesian trust models presented in [7], [16] the evidence that is used for calculating the trust value of another party is stored and processed by a reputation centre. In centralized models, it is possible to protect the users privacy by showing the users only an aggregated trust value or reputation value, without revealing who contributed what rating to the overall result. However, this requires a fully trusted centralized authority. Furthermore, this approach has the disadvantage is that the centralized authority can hardly decide whether a rating for an interaction is justified or not.
- **Distributed Models:** In trust models as provided in [9], [10], [14], [17] each entity collects its direct evidence, i.e. the outcomes of its past interactions locally. In order to overcome the lack of direct evidence between entities, the models support the exchange of recommendations. The trust value of an interaction partner, is then computed locally by the entity that is interested in evaluating the trustworthiness of an interaction partner taking into account the entity's direct evidence and the recommendations from other parties, which are weighted according to the trustworthiness of the recommenders. The advantages of the distributed approach are: *(i)* it does not require a trusted third party *per se*, and *(ii)* the recommenders can decide based on the requesting entity, whether they are willing to provide a recommendation, and *(iii)* the parties who are evaluating the recommendations can weight those recommendations based on the subjective trustworthiness that they assigns to the recommenders. However, this leads to the situation that in current approaches, the parties who receives a recommendation can learn about the previous interactions and the preferences of their recommenders.

C. Privacy in distributed systems

We see two basic approaches to preserve the users' privacy in distributed systems:

- **Identity- / Pseudonym-based approach:** Following this approach, one tries to decouple the real-world identity of a user from his history in the trust system by introducing pseudonyms [5], [18].
- **Encryption-based approach:** Following this approach, one tries to encrypt recommendations using a homomorphic crypto-system. Thus, the party who evaluates the trustworthiness of an entity, can still perform computation on the encrypted recommendations, but is not able to learn about the individual recommendations [19].

D. Using encryption to preserve privacy

In this paper, we use an encryption-based approach[3]. In the rest of this section we introduce some state of the art proposals and show why they are not fitting for the stated problem.

The approach presented in [19] uses homomorphic encryption. It does not allow an entity that receives a recommendation to be able to learn about the trustworthiness of the recommenders. Furthermore, the approach does not take into account on how much knowledge a

[2]In non-Bayesian models, an unknown entity usually has an initial trust value, which is arbitrarily assigned, e.g., 0 or 0.5. However, as for the entity receiving this value, it is not possible to decide whether it is based on experience or whether this value is just the initial trust value, those values would skew the computed trust value.

[3]The approach could be used complementary to the identity-based approach proposed in [5].

recommendation is based upon. Especially, in case that a recommender has not interacted with the interaction partner under evaluation before, it would answer with an initial trust value, which would skew the calculation of the trust value in the end.[4]

Another approach in this class was presented in [20]. This approach is, like ours, based on homomorphic encryption and zero-knowledge proofs. However, it does not support negative ratings.

There are several other approaches tackling the field of trust and privacy, e.g., [21], [22], but the approaches mentioned before are the closest to our solution.

III. Assessing Trust While Preserving Privacy

In the following, we propose a novel approach for calculating a trust value in a privacy preserving manner. The approach is based on a Bayesian trust model, homomorphic encryption, and zero-knowledge proofs. We introduce the basic mechanism for the evidence-based computation of trust, and afterwards, two protocols that support a privacy-friendly computation of trust.

A. Bayesian Representation

This section starts with a short introduction of a representation of trust which is based on the trust and reputation models presented in [7], [9], [10], [12], [14].

The main parameters used to derive the trustworthiness of an entity in the Bayesian representation are the numbers of positive r and negative s evidence that have been collected based on direct experience and recommendations. Furthermore, r_0 and s_0 indicate the prior knowledge (as in [7], [9], [10], [12], [23]). The corresponding Beta probability density function $f(p \mid \alpha, \beta)$ is defined as:

$$f(p \mid \alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\cdot\Gamma(\beta)} \cdot p^{\alpha-1} \cdot (1-p)^{\beta-1} \quad , \quad \text{where } 0 \le p \le 1,\ \alpha > 0,\ \beta > 0 \quad . \tag{1}$$

Given the parameters r_0 and s_0, which may be chosen context-dependent as proposed in [12] or set to $r_0 = s_0 = 1$ as in [7], [9], [10]. The opinion about the trustworthiness of an entity is denoted as $o = (r, s)$. The expectation value of an opinion is referred to as $E(o)$ or $E((r, s))$. It can be calculated as:

$$E(o) = E((r,s)) = \frac{r + r_0}{r + s + r_0 + s_0} \tag{2}$$

B. Computational Model of Trust

In the following, we refer to the consumers of recommendations and service providers as *entities*. The computational trust model provides means for combining the direct evidence of the initiator, i.e., the consumer that wants to select a service, and indirect evidence (recommendations) from third parties. This can also be referred to as trust propagation. The basic ideas for trust propagation in the proposed approach are similar to the ones presented in [7], [23]. Therefore, the operators for the trust propagation are given the same names. The *consensus* operator combines several opinions to a single one, and the *discounting* operator allows to weight recommendations based on the reputation of, or opinion about, the recommender.

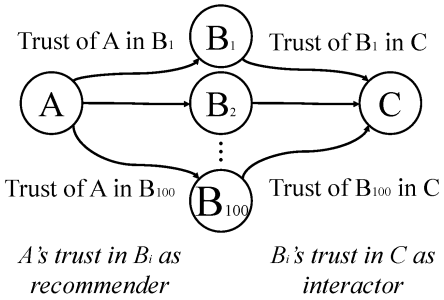

Figure 1. Trust network

For the explanation of the trust propagation a simple network is illustrated in Figure 1. Here, entity A is in the role of the initiator of an interaction, i.e., entity A has to select a service provider from a set of available service providers. As a basis for the selection, the initiator evaluates the trustworthiness of the candidates. In order to evaluate the trustworthiness of a candidate C, entity A uses its direct evidence and indirect evidence, which are also referred to as recommendations. In the example, entity A does not have any direct evidence, but it receives recommendations from the recommenders $B_1, B_2, \ldots, B_{100}$.

It is necessary to distinguish between the different contexts in which an entity gained trust. An entity may be trustworthy in the context of providing a service or in the context of providing recommendations about certain classes of service, as illustrated in Figure 1. The first is important for the selection of a candidate, the latter is important when deriving trust based on recommendations. An entity gains trust as service provider, when it provides a service that meets the expectation of the service consumer, and loses trust when it does not. Yet, the behavior of an entity when providing a service does not necessarily convey information about its behavior as recommender – and vice versa – as both contexts refer to different services offered by an entity. Therefore, trust is derived differently in both contexts. In the context of service provision, an entity obtains trust based on the fulfillment of consumers' expectations. In the context of providing recommendations an entity increments its trust when it provides accurate recommendations. In the following, we will assume that the trustworthiness of an entity in the context of providing recommendations is already given. For approaches to derive this information from the accuracy of previous recommendations see [10], [14].

Additionally, it is worth mentioning that the trustwor-

[4] A possible way around would require the redefinition of the recommenders set. However, such a change would heavily increase the computational costs of this proposal.

thiness in the context of providing recommendations as well as in the context of providing a service also depends on the considered class of service, e.g., file sharing or weather forecast. Yet, in the following we assume there is only one class of service, in order to keep the notation simple.

C. Notation

In the rest of the paper the following notation is used. A denotes initiators of interactions, i.e., service consumers. B denotes recommenders of services. C denotes service providers.

The opinion of an service consumer A about a service provider C is denoted as o_c^A (with a lowercase c). The opinion of a service consumer A about a recommender B is denoted as o_B^A (with an uppercase B).

The expectation value $E(o_c^A)$ express the trustworthiness that A assigns to C in the context of service provisioning. The expectation value $E(o_B^A)$ express the trustworthiness of A in B in the context of providing accurate recommendations.

D. Trust Propagation

For trust propagation, we define two basic operators: *consensus* and *discounting*.

Definition 3.1 (Consensus): Let $o_c^{B_1} = (r_c^{B_1}, s_c^{B_1})$ and $o_c^{B_2} = (r_c^{B_2}, s_c^{B_2})$ be the opinions of B_1 and B_2 about the trustworthiness of entity c. The opinion $o_c^{B_1,B_2} = (r_c^{B_1,B_2}, s_c^{B_1,B_2})$ is modeled as the opinion of an imaginary entity which made the experiences of B_1 and B_2, and is defined as:

$$o_c^{B_1,B_2} = o_c^{B_1} \hat{\oplus} o_c^{B_2} = (r_c^{B_1} + r_c^{B_2}, s_c^{B_1} + s_c^{B_2}) \tag{3}$$

Where the '$\hat{\oplus}$' symbol denotes the consensus operator. The operator can easily be extended for the consensus of multiple opinions (see Eq. 6).

Definition 3.2 (Discounting): Let $o_B^A = (r_B^A, s_B^A)$ and $o_c^B = (r_c^B, s_c^B)$. We denote the opinion of A about c based on the recommendation of B as $o_c^{A:B} = (r_c^{A:B}, s_c^{A:B})$ and define it as:

$$o_c^{A:B} = o_B^A \hat{\otimes} o_c^B = (d(o_B^A) \cdot r_c^B, d(o_B^A) \cdot s_c^B) \tag{4}$$

The discounting factor $d(o_B^A)$ can be defined as:

$$d(o_B^A) = E(o_B^A) = \frac{r_B^A + r_0}{r_B^A + s_B^A + r_0 + s_0} \tag{5}$$

Where the '$\hat{\otimes}$' symbol denotes the discounting operator. The aggregation of recommendations is done using the operators consensus and discounting. Assuming that A receives recommendations about C from a set of recommenders $\mathcal{B}$, where $\mathcal{B} = \{B_1, \ldots, B_n\}$. The trustworthiness that A assigns to $\mathcal{B}$ (in the context of providing recommendations) is given by $o_{B_1}^A, \ldots, o_{B_n}^A$. The recommendations are given as $o_c^{B_1}, \ldots, o_c^{B_n}$. The aggregation of the opinions using the operators defined above is calculated as follows:

$$\begin{aligned} o_c^{A:\mathcal{B}} &= (o_{B_1}^A \hat{\otimes} o_c^{B_1}) \hat{\oplus} \ldots \hat{\oplus} (o_{B_n}^A \hat{\otimes} o_c^{B_n}) \\ &= (\sum_{i=1}^{n} d(o_{B_i}^A) \cdot r_c^{B_i}, \sum_{i=1}^{n} d(o_{B_i}^A) \cdot s_c^{B_i}) \end{aligned} \tag{6}$$

Whenever entity A has additional direct evidence o_c^A, this evidence needs also be to considered, e.g., by adding this evidence after aggregating the recommendations or by considering oneself as recommender with a discounting factor $d = 1$.

E. Learning whom to trust

Assume that $f_c \in \{0, 1\}$ is A's feedback for the interaction – where '0' means "*not satisfying*" and '1' means "*satisfying*".A updates his direct evidence about C, $o_c^A = (r_c^A, s_c^A)$, after the interaction to $o_c^A = (r_c^A + f_c, s_c^A + 1 - f_c)$ – which means that $f_c = 1$ leads to an increase of r_c^A by 1 (while s_c^A remains unchanged) and $f_c = 0$ leads to an increase of s_c^A by 1 (while r_c^A remains unchanged).

After receiving recommendations from B_i, the update of the trustworthiness of the recommenders can be carried out based on the accuracy of the recommendation [24]. A recommendation is supposed to be accurate if the recommendation and the feedback f_c for outcome of an interaction have the same tendency. This can formalized as follows (f_i^h characterizes whether A rates the h-th recommendation by B_i as accurate):

- positive update $f_i^h = 1$: If feedback f_c is positive and the recommender provided more positive than negative evidence, i.e., $r_c^{B_i} > s_c^{B_i}$.
- negative update $f_i^h = 0$: If feedback f_c is negative and the recommender provided more negative than positive evidence, i.e., $r_c^{B_i} < s_c^{B_i}$.
- no update: If the recommender has no previous experience with C, i.e., $(r_c^{B_i}, s_c^{B_i}) = (0, 0)$. Or if the recommender provided as much negative as positive evidence, i.e., $r_c^{B_i} = s_c^{B_i}$.

Finally, the update will be calculated similar to the update of the trustworthiness of an interaction partner: If the trustworthiness of the recommender was $o_{B_i}^A = (r_{B_i}^A, s_{B_i}^A)$ before the update, it will be $(r_{B_i}^A + f_i^h, s_{B_i}^A + 1 - f_i^h)$ after the interaction.

IV. Goals for the privacy-friendly computation

Our approach for a privacy-friendly computation of trust has three goals:

1) *Support A in evaluating the trustworthiness of his potential interaction partners C using recommendations.* There is no need that A knows the individual recommendations as long as A can use them in calculations and compute the result. This goal is achieved using homomorphic encryption and a trusted third party that is trusted to perform a decryption.
2) *Support A in evaluating the trustworthiness of his recommenders B_i.* For A to update the trust in recommenders B_i (see Section III-E), A does not need

to know each individual recommendation $(r_c^{B_i}, s_c^{B_i})$, but one only needs to know whether the recommendation was accurate or not (thus, individual recommendations can be encrypted). For this evaluation it is sufficient to know if $r_c^{B_i} > s_c^{B_i}$, $r_c^{B_i} < s_c^{B_i}$, or $r_c^{B_i} = s_c^{B_i}$. However, A has to make sure that this information is reliably linked to the provided (encrypted) recommendation. This is achieved using zero-knowledge proofs.

3) *Prevent A from learning unnecessary details about the recommendations from his recommenders B_i.* Our evaluation shows that a curious, non-malicious, A will not be able to learn the values of a recommendation $(r_c^{B_i}, s_c^{B_i})$. Furthermore, we increase the costs for malicious attacker to learn unnecessary details about the recommendations when comparing our proposal against the state of the art.

V. Privacy-preserving Computation of Trust

In this section we introduce our novel protocol for a privacy-preserving computation of trust. For a better understanding, we divided the protocol into two parts. The first part introduces a preliminary protocol that can achieve the first two objectives stated in Section IV. The second part extends the preliminary protocol to achieve the final objective listed in Section IV. Details regarding homomorphic encryption and the used zero-knowledge proofs (ZKP) are presented in Section VII.

The protocol has the following parties and roles, which extends the notation presented in Section III-C.

- A is the initiator of interactions. A evaluates the trustworthiness of a service provider C based on the recommendations from recommenders B_i.
- Z is a trusted third party (TTP). Z's role is to decrypt data that is sent to it. Z is trusted by all protocol participants not to collude with A or B_i.

The protocol is divided into the following 4 phases:

1) *Phase 0.* The setup phase initializes the protocol parameters and define the set of participants.
2) *Phase 1.* A calculates the trustworthiness of service providers C.
3) *Phase 2.* For A to select the best service provider C.
4) *Phase 3.* For A to update the trust values regarding the service provider C and the recommenders B_i.

Each phase can be divided into one or more steps, where steps are named according to the following notation: <*Entity*>-<*Step Number for this Entity*>, where entity is either A, B or Z, and the step number $\in \mathbb{N}^*$.

A. Preliminary Protocol (Part 1)

The overview of the preliminary protocol is depicted in Figure 2.

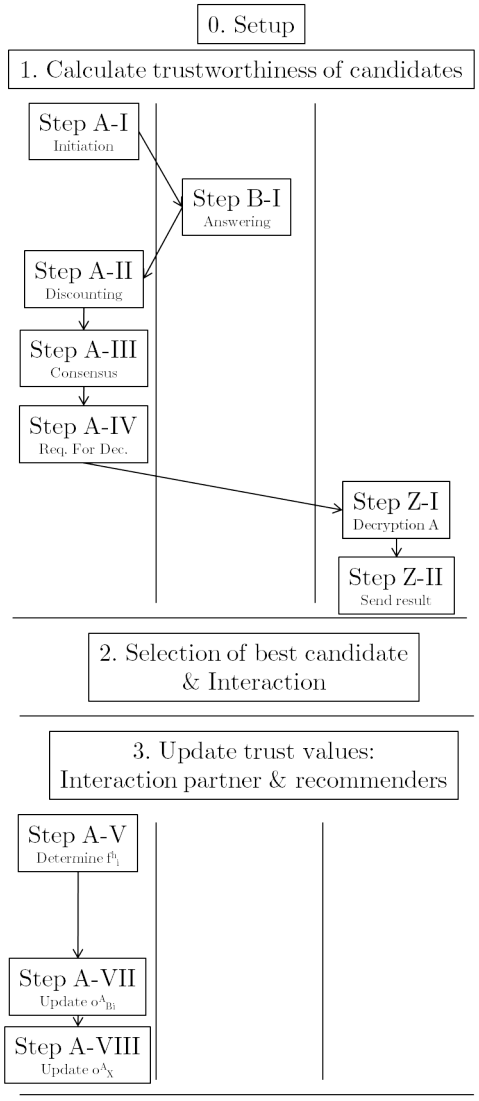

Figure 2. Overview: Preliminary Protocol – Part 1

0) Setup: The setup phase initializes protocol parameters, and define the set of participants.

- A defines a Sybil-free set $\mathcal{B} = \{B_1, \ldots, B_i, \ldots, B_n\}$ of entities that have agreed to provide recommendation regarding a service provider C. Additionally, the elements of $\mathcal{B}$ must agree that $|\mathcal{B}|$ is sufficiently large to be used as an anonymity set.[5]
- A initializes the opinions $o^A_{B_i}$ about the trustworthiness of his recommenders using the parameters $r_0 > 0$ and $s_0 > 0$, e.g., $r_0 = s_0 = 1$.
- A and all recommenders B_i in the recommender set agree on a trusted third party Z.
- A informs all recommenders B_i about N,[6] which is the maximum number of evidence A accepts from each recommender.

[5] A Sybil-free set of pseudonyms can be achieved as in [25], [26].

[6] N is necessary to prevent that an entity attacks the computation of trust by providing an arbitrary high number of pieces of evidence.

- Z creates a key pair with a public key (sk, pk) (see Section VII) and distributes the public key pk to A and all B_i. Z is trusted to keep sk secret.

1) Calculate Trustworthiness of Service Providers: When A has to evaluate the trustworthiness of a set of candidates for an interaction, then A initiates the following protocol for each candidate C.

Step A–I: Initiating a request for recommendations.

01: A sends a request for recommendation about C to all B_i.

Step B-I: Answering a request.

01: Each B_i responds with $(\mathsf{Enc}_{pk}(r_c^{B_i}), \mathsf{Enc}_{pk}(s_c^{B_i}))$ and the information needed to proof (using ZKPs) $(*)$:
  a) $0 \leq r_c^{B_i} + s_c^{B_i} \leq N$ and
  b) the true statement out of:
    b1) $r_c^{B_i} = s_c^{B_i}$
    b2) $r_c^{B_i} < s_c^{B_i}$
    b3) $r_c^{B_i} > s_c^{B_i}$

Step A–II: Processing recommendations (discounting).

01: For each reply by B_i
02: A verifies that the recommendation is valid (ZKP), i.e., $0 \leq r_c^{B_i}, s_c^{B_i} \leq N$ (using ZKPs) $(*)$
03: A calculates $\mathsf{Enc}_{pk}(r_c^{A:B_i}) = \mathsf{Enc}_{pk}(d(o_{B_i}^A) \cdot r_c^{B_i}) = d(o_{B_i}^A) \otimes \mathsf{Enc}_{pk}(r_c^{B_i})$ $(**)$
04: A calculates $\mathsf{Enc}_{pk}(s_c^{A:B_i}) = \mathsf{Enc}_{pk}(d(o_{B_i}^A) \cdot s_c^{B_i}) = d(o_{B_i}^A) \otimes \mathsf{Enc}_{pk}(s_c^{B_i})$ $(**)$
05: End for

Step A–III: Recommendation aggregation (consensus).

01: A computes $(**)$: $\mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} r_c^{A:B_i}) = \mathsf{Enc}_{pk}(r_c^{A:B_1}) \oplus \ldots \oplus \mathsf{Enc}_{pk}(r_c^{A:B_{|B|}})$
02: A computes $(**)$: $\mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} s_c^{A:B_i}) = \mathsf{Enc}_{pk}(s_c^{A:B_1}) \oplus \ldots \oplus \mathsf{Enc}_{pk}(s_c^{A:B_{|B|}})$

Step A–IV: Request for decryption.

01: A sends to Z $(***)$: $((\mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} r_c^{A:B_i}), \mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} s_c^{A:B_i}))$

Step Z–I: Decryption of message from A.

01: When Z receives a tuple $(\mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} r_c^{A:B_i}), \mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} s_c^{A:B_i}))$ from A, then
02: Z decrypts the tuple: $r_{agg}^A = \mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} r_c^{A:B_i}))$, and $s_{agg}^A = \mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} s_c^{A:B_i}))$

Step Z–II: Send result to A.

01: Z sends (r_{agg}^A, s_{agg}^A) to A

Important notes:

$(*)$ZKP techniques are explained in Section VII.

$(**)$Assuming a homomorphic encryption scheme as introduced in Section VII. We use the following short notation for operations when exploiting the homomorphism: $\mathsf{Enc}_{pk}(c \cdot m) = c \otimes \mathsf{Enc}_{pk}(m)$ and $\mathsf{Enc}_{pk}(m + m') = \mathsf{Enc}_{pk}(m) \oplus \mathsf{Enc}_{pk}(m')$

$(***)A$ can add his own experience to $(\mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} r_c^{A:B_i}), \mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} s_c^{A:B_i}))$ by encrypting $\mathsf{Enc}_{pk}(r_c^A, s_c^A)$ and treating it as another recommendation.

2) Selection of the Best Service Provider & Interaction: If there is only one candidate, A can use the collected information to decide whether this candidate is trustworthy enough for the interaction. If A can choose between multiple candidates, A runs the first part of the protocol per interaction partner. Afterwards, A can decide whether to interact with a certain interaction partner (e.g., the best one) or not. In case A decides not to interact, the protocol will stop without any further steps.

3) Update Trust Values: The update will only be carried out if A has interacted and rated the interaction with C and if B_i provided a (positive or negative) recommendation about C.

Step A–V: Determine f_i^h.

Assume that $f_c \in \{0, 1\}$ is A's feedback for the interaction. Having received a recommendation from B_i, A rates the accuracy of B_i's recommendation either positive or negative, i.e., $f_i^h = 1$ or $f_i^h = 0$ (where h denotes that A updates his trust in the recommender B_i for the h^{th} time).

01: For each B_i
02: A calculates the accuracy f_i^h of the recommendation as
(Knowledge about relation between $r_c^{B_i}$ and $s_c^{B_i}$ was provided in Phase B–I.)

$$f_i^h = \begin{cases} 1 & \textbf{if } f_c = 1 \text{ and } r_c^{B_i} > s_c^{B_i}, \\ 1 & \textbf{if } f_c = 0 \text{ and } r_c^{B_i} < s_c^{B_i}, \\ \text{undef} & \textbf{if } r_c^{B_i} = s_c^{B_i}, \\ 0 & \textbf{else.} \end{cases}$$

03: End for

Step A–VII: Update $o_{B_i}^A$.

01: For each B_i
02: If $f_i^h \neq \text{undef}$, then A updates $o_{B_i}^A = (r_{B_i}^A, s_{B_i}^A)$ to $o_{B_i}^A = (r_{B_i}^A + f_i^h, s_{B_i}^A + 1 - f_i^h)$
03: End for

Step A–VIII: Update o_c^A.

Assume that $f_c \in \{0, 1\}$ is A's feedback for the interaction.

01: A updates $o_c^A = (r_c^A, s_c^A)$ to $o_c^A = (r_c^A + f_c, s_c^A + 1 - f_c)$

4) Properties of the preliminary protocol: Assuming that all entities follow the protocol, we have that:

- A can calculate the trust value for C as described in Section III-B.
- A does not have direct access to the recommendations of B_i, as A gets only encrypted data from B_i.
- A can learn about the trustworthiness of B_i based on the information provided by the ZKP.
- Z does not have access to the individual recommendations, as it does not receive them, but only aggregated values.

5) Vulnerabilities of the preliminary protocol: A curious attacker could exploit the preliminary protocol using two different strategies, which require no preparation:

1) A can send any tuple to Z for decryption, e.g., $(r_c^{B_i}, s_c^{B_i})$ in step A–IV: As Z could not distinguish those from the correctly computed values, Z would decrypt those values and send them back to A.
2) A can select the values for $d_0, \ldots, d_n$ without any control. If A selects $d_i = 1$ and $\forall j \neq i.d_j = 0$, then it holds that $(r_{agg}, s_{agg}) = (r_c^{B_i}, s_c^{B_i})$ and A can get knowledge about an arbitrary recommendation.

There exists a number of options to overcome the first attack. However, the second attack is harder to handle, as it is inherent of the trust updating process that entity A controls the values of $d(o_{B_i}^A)$. The full protocol, which extends of the preliminary protocol handles both attacks. The full protocol is presented next.

B. Full Protocol (Part 2)

To overcome the vulnerabilities presented in the preliminary protocol, we extended it. This extended version, i.e., the full protocol, prevents A from manipulating the values of $d(o_{B_i}^A)$.

The full protocol has the same phases, but additional steps to phases 1 (for calculating the trustworthiness of service providers) and 3 (for updating trust values), and one suppressed step in phase 1 (step Z–II). Phase 0 (setup) has additional requirements as well. The full protocol is depicted in Figure 3.

The building blocks that differ from preliminary protocol are highlighted in gray in Figure 3.

0) Setup: The setup phase initializes protocol parameters, and define the set of participants. The first five items are common with the preliminary protocol, while the other items are added to support the extra functionalities of the full protocol.

- A defines a Sybil-free set $\mathcal{B} = \{B_1, \ldots, B_i, \ldots, B_n\}$ of entities that have agreed to provide recommendation regarding a service provider C. Additionally, the elements of $\mathcal{B}$ must agree that $|\mathcal{B}|$ is sufficiently large to be used as an anonymity set.
- A initializes the opinions $o_{B_i}^A$ about the trustworthiness of his recommenders, and sends $\mathsf{Enc}_{pk}(r_0)$ and $\mathsf{Enc}_{pk}(s_0)$ to B_i together with the information for a ZKP for: $r_0 + s_0 = 2$ and $r_0 > 0$ and $s_0 > 0$.
- A and all recommenders B_i in the recommender set agree on a trusted third party Z.
- A informs all recommenders B_i about N, that is the maximum number of evidence A would take from each recommender.
- Z distributes its public key pk to A and all B_i.
- A and all recommenders agree on a value for $r_0 + s_0$ for the opinions on the trustworthiness of the recommenders. For simplicity, we assume they agreed on $r_0 + s_0 = 2$.[7]
- All recommenders in the recommender set of A agree on two random functions $rand_1(a, b)$ and $rand_2(a, b)$, where $rand_1(a, b) = -rand_1(b, a)$ and $rand_2(a, b) = -rand_2(b, a)$.[8]
- The recommenders agree on a function *partner* which assigns each recommender exactly one partner. For simplicity, we assume that we have a even number of recommenders, and the function partner is defined in a way that it holds $partner(i) = j \Leftrightarrow partner(j) = i$.
- A initializes $o_{B_i}^A = (0, 0)$ for all B_i.

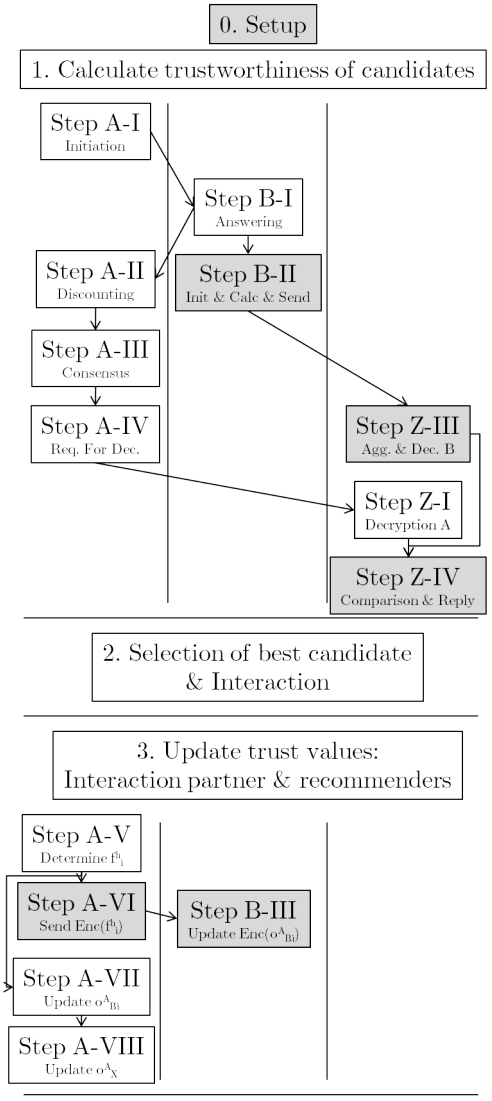

Figure 3. Overview: Full Protocol – Part 2

1) Calculate trustworthiness of candidates: When A has to evaluate the trustworthiness of a set of candidates for

[7]It would also be possible to choose another value for the sum of $r_0 + s_0$. The selection of the individual parameters $r_0 > 0$ and $s_0 > 0$ allows A to personalize the individual trustworthiness of his recommenders.

[8]The functions $rand_1$ and $rand_2$ have to be chosen in compliance with the parameters of the crypto-system introduced in Section VII.

interaction, A initiates the protocol shown in Figure 3.

Step B–II: B_i initializes, calculates and obfuscates its discounted recommendation and sends it to Z.

01: Whenever B_i receives $(\mathsf{Enc}_{pk}(f_i^h))$, then
02: If B_i has never run this protocol with A before, then B_i defines $\mathsf{Enc}_{pk}(r^A_{B_i}) = \mathsf{Enc}_{pk}(0)$ and $counter_A = 0$
03: End If
04: B_i calculates $\mathsf{Enc}_{pk}(r_c^{A:B_i}) = \mathsf{Enc}_{pk}(d(o^A_{B_i}) \cdot r_c^{B_i}) = = \frac{1}{counter_A+2} \otimes (\mathsf{Enc}_{pk}(r^A_{B_i}) \oplus \mathsf{Enc}_{pk}(r_0)) \otimes r_c^{B_i}$
05: B_i calculates $\mathsf{Enc}_{pk}(s_c^{A:B_i}) = \mathsf{Enc}_{pk}(d(o^A_{B_i}) \cdot s_c^{B_i}) = = \frac{1}{counter_A+2} \otimes (\mathsf{Enc}_{pk}(r^A_{B_i}) \oplus \mathsf{Enc}_{pk}(r_0)) \otimes s_c^{B_i}$
06: B_i sends the following values to Z: $\mathsf{Enc}_{pk}(r_c^{A:B_i}) \oplus \mathsf{Enc}_{pk}(rand_1(i, partner(i)))$ and $\mathsf{Enc}_{pk}(s_c^{A:B_i}) \oplus \mathsf{Enc}_{pk}(rand_2(i, partner(i)))$

Step Z–III: Aggregation and decryption of messages by all B_i.

01: When Z has received the discounted recommendations by all recommenders B_i in the set of recommenders, then
02: Z calculates $\sum_{i=1}^{|B|} \mathsf{Enc}_{pk}(r_c^{A:B_i}) \oplus \mathsf{Enc}_{pk}(rand_1(i, partner(i)))$
03: Z calculates $\sum_{i=1}^{|B|} \mathsf{Enc}_{pk}(s_c^{A:B_i}) \oplus \mathsf{Enc}_{pk}(rand_2(i, partner(i)))$
04: Z decrypts the results: $r^B_{agg} = \mathsf{Dec}_{sk}(\sum_{i=1}^{|B|} \mathsf{Enc}_{pk}(r_c^{A:B_i}) \oplus \oplus \mathsf{Enc}_{pk}(rand_1(i, partner(i)))$ and $s^B_{agg} = \mathsf{Dec}_{sk}(\sum_{i=1}^{|B|} \mathsf{Enc}_{pk}(s_c^{A:B_i}) \oplus \oplus \mathsf{Enc}_{pk}(rand_2(i, partner(i)))$

Step Z–IV: Comparison and reply.

01: If $r^A_{agg} = r^B_{agg}$ and $s^A_{agg} = s^B_{agg}$ Z sends (r^A_{agg}, s^A_{agg}) to A, else Z sends *error* to A.

2) Selection of the best recommender & Interaction:: This phase remains the same, as described in the preliminary protocol.

3) Update trust values: The following steps are added to this phase:

Step A–VI: A sends update to B_i.

01: For each B_i
02: If $f_i^h \neq undef$
03: A calculates $\mathsf{Enc}_{pk}(f_i^h)$
04: A generates the information for a ZKP showing $\mathsf{Enc}_{pk}(f_i^h)$ encrypts either 0 or 1.
05: A sends the information from 03 and 04 to B_i
06: End if
07: End for

Step B–III: B_i updates $o^A_{B_i}$.

01: Whenever B_i receives $(\mathsf{Enc}_{pk}(f_i^h))$, then
02: B_i verifies that $(\mathsf{Enc}_{pk}(f_i^h))$ is valid (ZKP) $(*)$
03: B_i updates $\mathsf{Enc}_{pk}(r^A_{B_i})$ using $\mathsf{Enc}_{pk}(r^A_{B_i}) = \mathsf{Enc}_{pk}(r^A_{B_i}) \oplus \mathsf{Enc}_{pk}(f_i^h)$
04: B_i updates $counter_A = counter_A + 1$

In case A wants to add its direct experience, A can also participate in the protocol as a recommender (with a discount value of 1, as mentioned in Section III-D).

VI. Proofs and Properties

A. Proving the functionality

Proof for $d(o^A_{B_i})$ computed by A is equivalent to $d(o^A_{B_i})$ computed by B:

If the protocol was executed correctly, the following statements Initialization and Update are true:

1) Initialization: In the **Setup phase**, A initializes $o^A_{B_i} = (0, 0)$. Furthermore, A and B_i agree on $r_0 + s_0 = 2$, and A sends $Enc(r_0)$ and $Enc(s_0)$ to B.

- Thus, A would calculate $d(o^A_{B_i}) = \frac{r^A_{B_i}+r_0}{r^A_{B_i}+s^A_{B_i}+r_0+s_0} = \frac{r^A_{B_i}+r_0}{r^A_{B_i}+s^A_{B_i}+2}$. During the initialization, A calculates $d(o^A_{B_i}) = \frac{r_0}{2}$.
- On the other hand (see step B–II), B initializes $\mathsf{Enc}_{pk}(r^A_{B_i}) = \mathsf{Enc}_{pk}(0)$ and $counter_A = 0$. Thus, B calculates $\mathsf{Enc}_{pk}(d(o^A_{B_i})) = \frac{1}{counter_A+2} \otimes (\mathsf{Enc}_{pk}(r^A_{B_i}) \oplus \mathsf{Enc}_{pk}(r_0)) = \mathsf{Enc}_{pk}(\frac{r_0}{2})$.

2) Update: When A and B_i run the protocol correctly, they will always update the parameters for the calculation of $d(o^A_{B_i})$ in a synchronized manner (see steps A–VI, A–VII and B–III in Fig. 3). In the case that $f_i^h = undef$ nothing will happen. In the other cases, i.e. $f_i^h = 0$ or $f_i^h = 1$, it can be shown that A would either increase $r^A_{B_i}$ or $s^A_{B_i}$ by 1. Furthermore, if $f_i^h \neq undef$ B would increase $counter_A$ by 1, and in the case that $f_i^h = 1$ B would also increase $r^A_{B_i}$ by 1, i.e., $\mathsf{Enc}_{pk}(r^A_{B_i} \oplus \mathsf{Enc}_{pk}(1)$.

Given that A and B_i use the same initialization as shown above, we assume at a certain point in time it holds that A would calculate $d(o^A_{B_i}) = \frac{r^A_{B_i}+r_0}{r^A_{B_i}+s^A_{B_i}+r_0+s_0}$ and B would calculate $\mathsf{Enc}_{pk}(d(o^A_{B_i})) = \frac{1}{counter_A+2} \otimes (\mathsf{Enc}_{pk}(r^A_{B_i}) \oplus \mathsf{Enc}_{pk}(r_0)) = \mathsf{Enc}_{pk}(\frac{r^A_{B_i}+r_0}{counter_A+2})$, with $counter_A = r^A_{B_i} + s^A_{B_i}$. Without a loss of generality, we assume there is an update with $f_i^h = 1$.

- Thus, the next time, A would calculate $d(o^A_{B_i}) = \frac{r^A_{B_i}+1+r_0}{r^A_{B_i}+1+s^A_{B_i}+r_0+s_0} = \frac{r^A_{B_i}+1+r_0}{r^A_{B_i}+1+s^A_{B_i}+2}$.
- On the other hand, B would calculate $\mathsf{Enc}_{pk}(d(o^A_{B_i})) = \frac{1}{counter_A+1+2} \otimes (\mathsf{Enc}_{pk}(r^A_{B_i} + 1) \oplus \mathsf{Enc}_{pk}(r_0)) = \mathsf{Enc}_{pk}(\frac{r^A_{B_i}+1+r_0}{counter_A+1+2}) = \mathsf{Enc}_{pk}(\frac{r^A_{B_i}+1+r_0}{r^A_{B_i}+s^A_{B_i}+1+2})$

■

Proof for $r^A_{agg} = r^B_{agg}$ and $s^A_{agg} = s^B_{agg}$:

We show just the equivalence of $r^A_{agg} = r^B_{agg}$; the proof for $s^A_{agg} = s^B_{agg}$ could be done analogously. If the protocol was executed correctly, the following statements are true:

1) From steps A–IV and Z–I: We have

$$r^A_{agg} = \mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(\sum_{i=1}^{|B|} r_c^{A:B_i})) = \sum_{i=1}^{|B|} r_c^{A:B_i} = \sum_{i=1}^{|B|} d(o^A_{B_i}) \cdot r_c^{B_i}$$

2) From steps B–II and Z–III: We have

$$r^B_{agg} = \mathsf{Dec}_{sk}(\sum_{i=1}^{|B|} \mathsf{Enc}_{pk}(r_c^{A:B_i}) \oplus \mathsf{Enc}_{pk}(rand_1(i, partner(i))) = \sum_{i=1}^{|B|} r_c^{A:B_i} + rand_1(i, partner(i)) = \sum_{i=1}^{|B|} r_c^{A:B_i} = \sum_{i=1}^{|B|} d(o^A_{B_i}) \cdot r_c^{B_i}$$

Based on the assumptions that

- $j = partner(i) \Leftrightarrow i = partner(j)$,
- $rand_1(i, partner(i)) = -rand_1(partner(i), i)$, and
- each recommender R_i has exactly one partner,

one can conclude that $\sum_{i=1}^{|B|} rand_1(i, partner(i)) = 0$, and thus it holds $r^B_{agg} = \sum_{i=1}^{|B|} r_c^{A:B_i}$. ∎

B. Properties of this protocol

Assuming all entities follow the protocol:

- A can calculate the trust value for C.
- A does not have direct access to the recommendations of the B_i's.
- A can learn about the trustworthiness of its recommenders using the information whether $r > s$, $r < s$, $r = s$, which was verified based using ZKPs.
- Z cannot learn the values of the opinion $o^A_{B_i}$ of A about its recommenders B_i nor the corresponding discounting factor $d(o^A_{B_i})$ (as Z only receives the aggregated (and encrypted) values $\mathsf{Enc}_{pk}(r^A_{agg}, s^A_{agg})$ from A and the values Z receives from B_i are obfuscated by random numbers). Especially, Z could not get this information by decrypting the received information from A or any / all B_i.
- Z cannot learn the values of the individual recommendations $o_c^{B_i}$ (as the recommendations are obfuscated by random numbers).
- A cannot adjust the values for $o^A_{B_i}$ as free as in the basic version of the protocol. Especially, the protocol enforces that A can only incrementally update the values of $d(^A_{B_i})$ after A received a recommendation by B_i as part of the protocol. Furthermore, the protocol assures that the update will be either 0 or 1 (using ZKPs). This prevents a curious attacker from learning the values $(r_c^{B_i}, s_c^{B_i})$ in "a single shot", without preparation. However, a "long run-attack" is still possible, as A can tailor the incremental updates in the way that he pushes the trustworthiness of one recommender towards 1 and the trustworthiness of the other recommenders towards 0. Compared to the state-of-the-art approaches the extended version provides an advantage nevertheless, as it severely increased the costs of this type of attack.

C. Attack based on linear equations

If A gets the values r_{agg} and s_{agg}, then A can try to calculate the values for $r_c^{B_i}$ and $s_c^{B_i}$ based on repeated interactions. For example, for r_{agg} holds that $r_{agg} = \sum_{i=1}^{n} d(o^A_{B_i}) \cdot r_c^{B_i}$. As A knows the values for $d(o^A_{B_i})$ (and how they changed over time), A could learn all the values $r_c^{B_i}$ after n interactions, in the case that the values of $r_c^{B_i}$ have not changed over time.

In order, to overcome this problem, there are multiple solutions: *(i)* the recommenders B_i could introduce noise on the values $r_c^{B_i}$ and $s_c^{B_i}$, or *(ii)* Z could introduce noise the result r_{agg} and s_{agg}. Z can introduce imprecision in the following way: instead of r_{agg} and s_{agg} Z sends to A the value of $\frac{r_{agg}+r_0}{r_{agg}+s_{agg}+2}$ and an additional level of certainty lc, e.g., $lc = 1$ if $r_{agg} + s_{agg} \in [0; |B| \cdot N/3]$, $lc = 2$ if $r_{agg} + s_{agg} \in [|B| \cdot N/3 + 1; |B| \cdot 2N/3]$, and $lc = 3$ if $r_{agg} + s_{agg} \in [|B| \cdot N2/3 + 1; |B| \cdot N]$.

Finally, we can state that one could overcome the need for a central trusted third party, by distributing the key for the decryption to multiple entities and using threshold cryptography.

VII. Computing Trust Levels Privately

In the following section, we show how homomorhic encryption and zero-knowledge proofs that are used in the previously presented protocols can be realized.[9]

A public-key encryption scheme $\mathcal{E} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ consists of three efficient algorithms where KGen on input the security parameter 1^n (in unary) outputs a pair of private and public key $(sk, pk) \leftarrow \mathsf{KGen}(1^n)$, the encryption algorithm Enc on input pk and a message m (from some message space $\mathcal{M}_{pk}$ determined by pk) returns a ciphertext $C \leftarrow \mathsf{Enc}(pk, m)$, and the decryption algorithm Dec on input sk and C returns a message or a special symbol $\perp$ to indicate a decryption error. We assume that for any $(sk, pk) \leftarrow \mathsf{KGen}(1^n)$ and any message $m \in \mathcal{M}_{pk}$ we have $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m$, i.e., encrypted messages can be correctly decrypted. We occasionally specify the randomness ρ from the space $\mathcal{R}_{pk}$ used to derive the ciphertext and write $C = \mathsf{Enc}(pk, m; \rho)$ for this (now deterministic) process.

An encryption scheme $\mathcal{E}$ is *homomorphic* if there exists an operation $\otimes$ such that for any $(sk, pk) \leftarrow \mathsf{KGen}(1^n)$, any m, m' in the message space $\mathcal{M}_{pk}$ forming a group under operation $\oplus$, any randomness ρ, ρ' from $\mathcal{R}_{pk}$ forming a group under $\odot$ we have

$$\mathsf{Enc}(pk, m; \rho) \otimes \mathsf{Enc}(pk, m'; \rho') = \mathsf{Enc}(pk, m \oplus m'; \rho \odot \rho').$$

[9]The application of the cryptographic primitives requires a discretization of the trust values; it should be sufficient to consider 2 digits.

Note that we assume that ciphertext operation simultaneously complies with the group operations for messages and randomness; this will ensure that we can prove properties about encrypted values in zero-knowledge. We also assume that all group operations and inverse computations can be performed efficiently. Examples of such homomorphic encryption schemes include the ElGamal scheme [27], being homomorphic for messages over the underlying multiplicative group, and Paillier's scheme [28] which is homomorphic over $(\mathbb{Z}_N, +)$ for messages and $(\mathbb{Z}_N^*, \cdot)$ for randomness for RSA modulus N.

For Paillier's scheme a ciphertext is given by $C = g^m \rho^N \bmod N^2$ for $pk = (g, N)$ and randomness $\rho \in \mathbb{Z}_N^*$, and multiplication of ciphertexts $C = g^m \rho^N \bmod N^2$ and $C' = g^{m'}(\rho')^N \bmod N^2$ over $\mathbb{Z}_{N^2}^*$ yields

$$C \cdot C' = g^m \rho^N \cdot g^{m'}(\rho')^N = g^{m+m'}(\rho\rho')^N \bmod N^2$$

and thus an encryption of $m + m' \bmod N$ for randomness $\rho\rho' \in \mathbb{Z}_N^*$. One can also raise a ciphertext $C = g^m \rho^N \bmod N^2$ to a constant power $\alpha \in \mathbb{Z}_N$, yielding a ciphertext $C^\alpha = g^{\alpha m} \rho^{\alpha N} \bmod N^2$ of $\alpha m \bmod \mathbb{Z}_N$ for $\rho^\alpha \in \mathbb{Z}_N^*$. Hence, this scheme is well suited for our purpose since we are interested in sums of evidences.

We also assume that the homomorphic encryption scheme is *indistinguishable under chosen-plaintext attacks* (IND-CPA) [29] which roughly means that no efficient adversary can distinguish between encryptions of (known) messages m and m' under pk. We again note that Paillier's scheme is IND-CPA under the decisional composite residuosity assumption [28].

For simplicity we assume a trusted entity to hold the secret key sk and to provide decryptions when necessary. Using a threshold version of Paillier's scheme [30] we can distribute the secret key among several entities. To decrypt a ciphertext each entity then provides a partial decryption which the receiver can verify and from which the receiver can reconstruct the message. A system parameter determines how many corrupt decryption entities the scheme can tolerate such that security and robustness (i.e., the ability to decrypt correctly even if some parties provide malicious partial decryptions) are still guaranteed.

We note that in both cases, single authority and threshold version, the ability to submit arbitrary ciphertexts to the authorities enables malicious parties in principle to mount chosen-ciphertext attacks (akin to the vulnerabilities described in Section V-A). This is inherent when using the homomorphic encryption property to ensure private computations among parties (see also [31]), and is somewhat countered by the augmented protocol.

A. Proving Relationships

Using encrypted data introduces the problem of robustness in the sense that malicious parties may now unnoticeably enter, say, out-of-range values into the computation which would then dominate the sum over all values. To prevent such misbehavior we have the parties provide correctness proofs of their encrypted values. Such proofs on one hand protect an honest verifier against malicious provers, trying to convince the verifier about the validity of a false statement (soundness), and on the other hand ensures that a malicious verifier cannot learn anything about the evidence value of an honest prover beyond the values' validity (zero-knowledge).

We assume that all evidence values are bounded such that the sums remain smaller than the message space. This is justified by the aging of evidence, which typically keeps the values in the range of a few bits only, and vice versa justifies the deployment of aging. More formally, we assume that each party holds encryptions $R = \mathsf{Enc}(pk, r; \rho)$ and $S = \mathsf{Enc}(pk, s; \sigma)$ of the positive and of the negative evidence (for known randomness ρ and σ). We assume that r and s can be represented with k bits each and that $2^k \ll N$ for the underlying additive group of public order N.

There are essentially four cases:

1) If $r = s = 0$, i.e., the recommender cannot provide any evidence, then the recommender should be able to prove that the encrypted values are both 0. However, since such a correctness proof reveals r and s anyway, the recommender may simply communicate this in clear and let the receiver act accordingly.
2) For $r = s \neq 0$ the recommender should be able to prove equality of the values and to show that they are unequal to 0, but without revealing the actual values of r and s.
3) For $r > s$ the recommender should be able to prove that r, s are in the admissible range and that one exceeds the other (but without revealing the difference for example).
4) The case $s > r$ is analogous to the previous case.

a) Fundamental Proof Techniques.: We discuss some basic proof techniques. All proofs starts from the assumption that the prover holds an encryption $X = \mathsf{Enc}(pk, x; \xi)$ for some x, ξ. Then prover and verifier engage in a three-move (honest-verifier zero-knowledge) proof about some property about x. All these proofs share the same structure: the prover in the first encrypts some random value and sends the ciphertext to the verifier who replies with a random challenge c from a set $\mathcal{C} \subseteq \mathbb{Z}_N$. The prover finally sends a reply to the challenge which the verifier checks. The proof can be made non-interactive via the Fiat-Shamir transformation [32], [33] by hashing the commitment locally to derive the challenge; the proof is then zero-knowledge (even against malicious verifiers) in the random oracle model.

As an example to prove in zero-knowledge that $x = 0$ the prover first computes $C = \mathsf{Enc}(pk, 0; \gamma)$ and sends it to the verifier, the verifier sends c, and the prover replies with $\delta = \gamma \cdot \xi^c$. The verifier accepts iff $CX^c = \mathsf{Enc}(pk, 0; \delta)$. The proof is honest-verifier zero-knowledge because given X (but not x, ξ) the simulator picks c, γ at random, sets $C = \mathsf{Enc}(pk, 0; \gamma) \cdot X^{-c}$, and outputs $C, c, \delta = \gamma$ as the

transcript. Note that the simulated transcript is identically distributed to the one of an actual execution (with an honest verifier picking the challenge randomly), and could also be carried out if the random challenge is given to the simulator. We denote such an elementary proof by $\mathcal{ZK}\left(pk, \mathsf{Enc}(pk, x; \xi) : x = 0\right)$.

Given the basic proof for showing that $x = 0$ we can derive more complex proofs. For example, we can prove the logical AND of all proofs easily. Namely, the prover sends the first message for all proofs simultaneously and the verifier sends a single random challenge which the prover uses in all proofs to compute the replies. We can interpret this as a "big" proof $\mathcal{ZK}\left(pk, X_1 \wedge X_2 \wedge \cdots : P_1 \wedge P_2 \wedge \ldots\right)$ for the AND that all ciphertexts X_i satisfy properties P_j simultaneously (but where some property P_j may not depend on all X_i). The effort is linear in the number of underlying elementary proofs.

Another important logical statement is the OR of two statements [34], letting $\mathcal{ZK}\left(pk, X : P_1 \vee P_2 \vee \ldots\right)$ denote the proof that X satisfies at least on of the properties $P_1, P_2, \ldots$. Proving for example that $x \in \{0, 1\}$ for the ciphertext X, i.e., that $x = 0$ or that $x = 1$, boils down to show that X or $X \cdot \mathsf{Enc}(pk, -1; 1)$ for constant $\mathsf{Enc}(pk, -1; 1)$ encrypts 0. To this end the prover preselects a random challenge $c_{1-x} \leftarrow \mathcal{C}_{pk}$ and runs the zero-knowledge simulator (for challenge c_{1-x}) to create a simulated proof $(C_{1-x}, c_{1-x}, \delta_{1-x})$ that the part $X \cdot \mathsf{Enc}(pk, -1; 1)$ (for $x = 0$) resp. the part X (for $x = 1$) encrypts 0. For the other part X (for $x = 0$) resp. the part $X \cdot \mathsf{Enc}(pk, -1; 1)$ (for $x = 1$) it simply runs the prover for showing that the value indeed encrypts 0 where the challenge c_x is computed as $c_x = c - c_{1-x} \bmod N$ for the preselected challenge c_{1-x} and the verifier's random challenge c. For transcript $(C_0, C_1, c, c_0, \delta_0, c_1, \delta_1)$ the verifier checks that both proofs (C_0, c_0, δ_0) and (C_1, c_1, δ_1) are valid and that $c = c_0 + c_1 \bmod N$. Hence, the proof for showing $x \in \{0, 1\}$ roughly consists of two elementary proofs.

Given the proofs for logical statements AND and OR we can devise an (honest-verifier) zero-knowledge proof that $x \in [0, 2^k - 1]$. The proof is carried out by having the prover compute encryptions $X_i = \mathsf{Enc}(pk, x_i; \xi_i)$ of the individual bits x_i of $x = \sum_{i=0}^{k-1} x_i 2^i$. Then the prover shows that each X_i encrypts a bit and that $X \cdot \prod_{i=0}^{k-1} X_i^{2^{-i}}$ encrypts 0. Formally, the prover shows

$$\mathcal{ZK}\left(pk, X, X_0, \ldots, X_{k-1} : x = \sum_{i=0}^{k-1} x_i 2^i \wedge \bigwedge_{i=0}^{k-1} (x_i = 0 \vee x_i = 1)\right).$$

The overall effort corresponds to k encryptions and $2k + 1$ elementary proofs. We can now also prove easily that $x \in [B, B + 2^k - 1]$ for any constant B by proving that $X \cdot \mathsf{Enc}(pk, -B; 1)$ encrypts a value in $[0, 2^k - 1]$.

b) Proving relationships of evidence values.: The recommender, each time when passing on recommendationsproves that the ciphertexts of R and S encrypt values r, s such that

- $r = s$ by running
$$\mathcal{ZK}\left(pk, RS^{-1}, R, S : r - s = 0 \wedge r \in [0, 2^k - 1]\right) \wedge s \in [0, 2^k - 1];$$
- $r > s$ by running
$$\mathcal{ZK}\left(pk, RS^{-1}, R, S : r - s \in [1, 2^k] \wedge r \in [0, 2^k - 1] \wedge s \in [0, 2^k - 1]\right);$$
- $r < s$ by running
$$\mathcal{ZK}\left(pk, SR^{-1}, R, S : s - r \in [1, 2^k] \wedge r \in [0, 2^k - 1] \wedge s \in [0, 2^k - 1]\right).$$

Computationally, the most expensive cases are the ones for $r > s$ resp. $s > r$, requiring approximately $3k$ encryptions and $6k+3$ elementary proofs. Recall that k is small, usually in the order of 5, such that for Paillier's scheme the proofs can be performed with a few modular exponentiations only.

VIII. Conclusion

In this paper, we presented a novel, privacy-friendly approach of computing trust. To the best of our knowledge, this approach is the first one to combine the following features:

1) It supports entities in evaluating the trustworthiness of his potential interaction partners using recommendations.
2) It supports entities in evaluating the trustworthiness of his recommenders.
3) It prevents the curious initiators from learning unnecessary details about the preferences of his recommenders and it increases the costs of the attack by malicious attackers compared to the state of the art models.
4) It is able to handle negative evidence and absence of evidence.

Additionally, the privacy of the recommenders could be further strengthened if the recommender hide their identities using (Sybil-free) pseudonyms as proposed in [5].

References

[1] C. Dellarocas, "Online reputation systems: How to design one that does what you need." *Sloan Management Review*, vol. 51 (3), pp. 33–38, 2010.

[2] A. F. Westin, *Privacy and Freedom.* New York, NY, USA: Atheneum, 1967.

[3] S. Fischer-Hübner, *IT-Security and Privacy – Design and Use of Privacy-Enhancing Security Mechanisms*, ser. Lecture Notes in Computer Science. Springer-Verlag Berlin/Heidelberg, 2001, vol. 1958.

[4] L. A. Martucci, "Identity and anonymity in ad hoc networks," Ph.D. dissertation, Karlstad University, Jun 2009.

[5] L. Martucci, S. Ries, and M. Mühlhäuser, "Identifiers, privacy and trust in the internet of services," in *Proceedings of the 4th IFIP International Conference on Trust Management (IFIPTM 2010)*, 2010.

[6] G. Zacharia, A. Moukas, and P. Maes, "Collaborative reputation mechanisms in electronic marketplaces," in *HICSS '99: Proceedings of the Thirty-second Annual Hawaii International Conference on System Sciences-Volume 8.* Washington, DC, USA: IEEE Computer Society, 1999, p. 8026.

[7] A. Jøsang and R. Ismail, "The beta reputation system," in *Proceedings of the 15th Bled Conference on Electronic Commerce*, 2002.

[8] A. Jøsang, X. Luo, and X. Chen, "Continuous ratings in discrete bayesian reputation systems," in *2nd Joint iTrust and PST Conference on Privacy, Trust Management and Security*, Y. Karabulut, J. C. Mitchell, P. Herrmann, and C. D. Jensen, Eds. Springer Boston, 2008, pp. 151–166.

[9] S. Buchegger and J.-Y. Le Boudec, "A Robust Reputation System for Peer-to-Peer and Mobile Ad-hoc Networks," in *P2PEcon 2004*, 2004.

[10] W. T. L. Teacy, J. Patel, N. R. Jennings, and M. Luck, "Travos: Trust and reputation in the context of inaccurate information sources," *Autonomous Agents and Multi-Agent Systems*, vol. 12, no. 2, pp. 183–198, 2006.

[11] S. Ries, "CertainTrust: A trust model for users and agents," in *Proceedings of the 2007 ACM Symposium on Applied Computing*. ACM Press, 2007, pp. 1599 – 1604.

[12] S. Ries, "Extending bayesian trust models regarding context-dependence and user friendly representation," in *Proceedings of the 2009 ACM Symposium on Applied Computing*. New York, NY, USA: ACM, 2009, pp. 1294–1301.

[13] Z. Despotovic and K. Aberer, "Probabilistic Prediction of Peers' Performances in P2P Networks," *Engineering Applications of Artificial Intelligence*, vol. 18, no. 7, pp. 771–780, 2005, elsevier, 2005.

[14] S. Ries and A. Heinemann, "Analyzing the robustness of CertainTrust," in *2nd Joint iTrust and PST Conference on Privacy, Trust Management and Security*, Y. Karabulut, J. C. Mitchell, P. Herrmann, and C. D. Jensen, Eds. Springer, 2008, pp. 51 – 67.

[15] D. Gambetta, "Can we trust trust?" in *Trust: Making and Breaking Cooperative Relations*, D. Gambetta, Ed. New York: Basil Blackwell, 1990, pp. 213–237. [Online]. Available: http://www.sociology.ox.ac.uk/papers/gambetta213-237.pdf

[16] A. Whitby, A. Jøsang, and J. Indulska., "Filtering out unfair ratings in bayesian reputation systems," *The ICFAIN Journal of Management Research*, vol. 4(2), pp. 48 – 64, 2005.

[17] D. Quercia, S. Hailes, and L. Capra, "B-Trust: Bayesian trust framework for pervasive computing." in *4th International Conference on Trust Management (iTrust)*, ser. Lecture Notes in Computer Science, K. Stølen, W. H. Winsborough, F. Martinelli, and F. Massacci, Eds., vol. 3986. Springer, 2006, pp. 298–312.

[18] 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, Jun 2011.

[19] N. Gal-Oz, N. Gilboa, and E. Gudes, "Schemes for privately computing trust and reputation," in *Proceedings of 4th IFIP International Conference on Trust Management (IFIPTM 2010)*, ser. IFIP Advances in Information and Communication Technology, M. Nishigaki, A. Jøsang, Y. Murayama, and S. Marsh, Eds., vol. 321. Springer Boston, 2010, pp. 1–16.

[20] J. Bethencourt, E. Shi, and D. Song, "Signatures of reputation: Towards trust without identity (extended abstract)," in *Financial Cryptography*, 2010.

[21] M. Kinateder and S. Pearson, "A privacy-enhanced peer-to-peer reputation system," in *Proceedings of the 4th International Conference on Electronic Commerce and Web Technologies (EC-Web 2003)*, ser. LNCS, K. Bauknecht, A. M. Tjoa, and G. Quirchmayr, Eds., vol. 2738. Prague, Czech Republic: Springer-Verlag, September 2003, pp. 206–215. [Online]. Available: http://citeseer.ist.psu.edu/kinateder03privacyenhanced.html

[22] S. Steinbrecher, "Design options for privacy-respecting reputation systems within centralised internet communities," in *Security and Privacy in Dynamic Environments, Proceedings of the IFIP TC-11 21st International Information Security Conference (SEC 2006), 22-24 May 2006, Karlstad, Sweden*, ser. IFIP, S. Fischer-Hübner, K. Rannenberg, L. Yngström, and S. Lindskog, Eds., vol. 201. Springer, 2006, pp. 123–134.

[23] A. Jøsang, "A logic for uncertain probabilities." *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, no. 3, pp. 279–212, 2001.

[24] S. Ries, "Trust in ubiquitous computing," Ph.D. dissertation, Technische Universität Darmstadt, 2009.

[25] C. Andersson, M. Kohlweiss, L. A. Martucci, and A. Panchenko, "A Self-Certified and Sybil-Free Framework for Secure Digital Identity Domain Buildup," in *Information Security Theory and Practices: Smart Devices, Convergence and Next Generation Networks, Proceedings of the 2nd IFIP WG 11.2 International Workshop (WISTP 2008)*, ser. Lecture Notes in Computer Science, LNCS 5019. Springer, 13–16 May 2008, pp. 64–77.

[26] L. A. Martucci, M. Kohlweiss, C. Andersson, and A. Panchenko, "Self-Certified Sybil-Free Pseudonyms," in *Proceedings of the 1st ACM Conference on Wireless Network Security (WiSec'08)*. ACM Press, Mar 31 – Apr 2 2008, pp. 154–159.

[27] T. E. Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *CRYPTO'84*, ser. Lecture Notes in Computer Science, vol. 196. Springer-Verlag, 1985, pp. 10–18.

[28] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT'99*, ser. Lecture Notes in Computer Science, vol. 1592. Springer-Verlag, 1999, pp. 223–238.

[29] S. Goldwasser and S. Micali, "Probabilistic encryption." *J. Comput. Syst. Sci.*, pp. 270–299, 1984.

[30] P.-A. Fouque, G. Poupard, and J. Stern, "Sharing decryption in the context of voting or lotteries," in *Financial Cryptography 2000*, ser. Lecture Notes in Computer Science, vol. 1962. Springer-Verlag, 2001, pp. 90–104.

[31] P.-A. Fouque and D. Pointcheval, "Threshold cryptosystems secure against chosen-ciphertext attacks," in *ASIACRYPT*, ser. Lecture Notes in Computer Science, vol. 2248. Springer-Verlag, 2001, pp. 351–368.

[32] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *CRYPTO'86*, ser. Lecture Notes in Computer Science, vol. 263. Springer-Verlag, 1987, pp. 186–194.

[33] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *ACM Conference on Computer and Communications Security*, 1993, pp. 62–73.

[34] R. Cramer, I. Damgård, and B. Schoenmakers, "Proofs of partial knowledge and simplified design of witness hiding protocols," in *CRYPTO*, ser. Lecture Notes in Computer Science, vol. 839. Springer-Verlag, 1994, pp. 174–187.