Social Trust in Opportunistic Networks

Sacha Trifunovic and Franck Legendre
Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland
{lastname}@tik.ee.ethz.ch

Carlos Anastasiades
Institute of Computer Science and Applied Mathematics
University of Bern, Switzerland
anastasi@iam.unibe.ch

***Abstract*—Opportunistic networks enable mobile users to participate in various social interactions with applications such as content distribution and micro-blogs. Because of their distributed nature, securing user interactions relies rather on trust than hard cryptography. Trust is often based on past user interactions such as in reputation systems relying on ratings. Yet, a more fundamental trust, social trust – assessing a user is genuine with honest intentions – must be established beforehand as many identities can be created easily (i.e., sybils). By leveraging the social network structure and its dynamics (conscious secure pairing and wireless contacts), we propose two complementary approaches for social trust establishment: explicit social trust and implicit social trust. Complexity, trust propagation and security issues are evaluated using real world complex graphs, synthetic mobility models and mobility traces. We show how our approach limits the maximum number of sybils independently of the network size and is more robust against manipulation attacks compared to state-of-the-art approaches such as PGP-like certification chains and distributed community detection algorithms.**

I. Introduction

Opportunistic networks will change the way people communicate by allowing direct one-hop communications between handheld devices carried by human beings while on the move. Users will be involved in participatory interactions with their surrounding using applications (e.g., mobile social networking, content distribution [1], flea-markets, micro-blogs) enhancing the experience of real-world social networks with digital and ubiquitous features. With these applications, users will publish their input or services (e.g. content, sold objects, blog entries) and subscribe based on their solicitations. Inputs will disseminate from their authors to consumers through relays in a delay-tolerant epidemic fashion from hop to hop using mobility without routing per se. While areas of operations are mainly developing countries, for no fixed wireless infrastructure is required, urban citizens will also enjoy a free and open network that made the success of the Internet at its early stage.

In such an open environment where no central authority can be assumed, infrastructure-based and hard cryptographic solutions are often traded for threshold cryptography [2] or PGP-like chains [3]. Another prevailing solution used to secure interactions between possibly unknown users is trust. For instance, it is often considered in recommendation systems based on ratings, where trust relies on (i) the service (or content) quality provided by others and (ii) trust in other users' opinions having similar taste. This trust, however, requires interactions between users in order to be established. What is more, pure opportunistic networks cannot ensure a one-to-one binding between an identity and a user. Compared to real-world social networks, their digital counterpart allow to easily generate fake identities, known as sybil users [4]. These sybils can then obtain a higher influence in the system. Trust must hence be considered at a more fundamental level.

In this paper, we consider the most basic level of trust that can and must be achieved in opportunistic networks, i.e. *social trust*: the belief that an identity is genuine and that the user's intentions are honest. By leveraging the social network structure and its dynamics (i.e., secure pairing, wireless contacts [5]), we propose two approaches for social trust establishment that are robust to sybil attacks: *explicit social trust* and *implicit social trust*. We furthermore argue that neither PGP-like certificate chains nor distributed community detection algorithms [6] are suited to achieve this.

Explicit social trust is based on consciously established friend ties by building a robust tree-like graph of paired users. In contrast to PGP and Capkun et al.'s approach [7], which assume unconditional transitivity of trust, we calculate trust as a function of hop distance and interconnection resulting in decreasing trust with increasing hop distance and higher trust in users that are well connected in the resulting graph. Explicit social trust conveys trust that the identity is not sybil and verifies the honesty of the user's intentions since he or she paired consciously and can thus be easily detected and identified if misbehaving.

Implicit social trust leverages mobility properties using complex network tools, since one might not pair with every encountered user (e.g., some friends or familiar strangers). It builds another graph up to two-hops based on the familiarity of surrounding peers (i.e., the accumulated time of being in proximity) and the similarity (i.e. the amount of common familiars) to reinforce trust in a user. Implicit social trust conveys trust in the originality of identities due to their persistency, i.e. not being fast switching as sybils. Trust in the honesty of the user's intentions is not explicitly captured, but again, since the identity is persistent a misbehaving user (possibly sybil) would be easily spotted and punished.

Section II presents related work. Section III presents our social trust establishment algorithms, which are evaluated in Section IV in terms of complexity, dynamics and resilience against compromised nodes generating sybils. Section V discusses how to further secure our approach with prospects for further investigations. Section VI concludes our work.

II. Related Work

A sybil attack [4] describes the attempt to create many identities in order to gain larger influence in a reputation system, abandon bad reputation or evade responsibility of his/her actions. In order to detect such attacks, Piro et al. [8] observe

that sybil users can only communicate serially and thus cause much fewer collisions at the MAC layer. SybilGuard [9] considers that sybil users have only a few trust relationships which can be highlighted by carefully observing the social graph. Location-based sybil detection is also an effective measure [10] but requires specialized hardware. Note that all these approaches only provide a probabilistic assessment of a node being sybil.

Reputation systems are an ideal target for sybil attacks [11], [12]. These systems rely on disseminated user ratings to allow for an informed selection of content by estimating a prospective source's reputation beforehand. Liars or sybil liars try then to influence ratings in the system about a user or a service. The similarity of direct and received ratings may be evaluated to assess trust in future opinions [13], [14]. To avoid the manipulation of ratings, Quercia et al. propose to store them in tamper-proof tables certified by witnesses [15], [16]. Note that all these approaches rely their trust at the rating level, which requires interactions in the first place, while our approach focuses on a lower level of trust, social trust.

Since one cannot prevent users from generating multiple identities, one way to limit the influence of sybils is to proactively establish trust in the identities being genuine. In classical networks, trust is established by a certificate authority (CA) through a public key infrastructure (PKI) [17]. In a pure opportunistic network this approach is useless since no fixed infrastructure and thus no authorities can be assumed. The CA duty can, however, be distributed to nodes which can generate their own credentials and sign certificates of others when paired. Following this track, Capkun et al. [18] allows users to build certificate chains similar to PGP under the assumption of unconditional transitivity of trust along the chain paths. Other approaches limit trust exclusively to consciously selected friends [7] (non-transitive) or small groups [19]. Our approach relies on both friend ties and conditional transitivity of trust, depending on hop distance and social interconnection.

Besides crypto-related approaches, trust establishment can leverage mobility properties and network structures using the rich set of complex social network tools. For instance, community detection algorithms extract the underlying structure with the highest modularity when fed with a network topology [20], [21], [22]. Distributed versions for opportunistic networks such as proposed by Hui et al. [6], [23] classify users in different categories i.e., friends, familiar strangers, and strangers. Each category can be assigned different trust values e.g., to choose trustworthy forwarders in DTNs. This approach, however, defines strict categories and was not designed with security in mind especially against sybil attacks. This is why we claim that community detection algorithms are not suited for trust establishment and propose a novel approach next.

III. Social Trust Establishment

In the following, we propose two kinds of complementary social trust, *explicit* and *implicit*, and how to combine them.

A. From Friend Ties to Explicit Social Trust

The central elements of explicit social trust are consciously selected friend ties. Due to the mobility of the devices, users can establish secure and reliable friend ties whenever they meet by secure pairing. In contrary to PGP and Capkun et al. [18], we assume conditional transitivity of trust[1] depending on hop distance and connectivity, i.e. trust in a user connected to several common friends is higher than in a user with one single connection over various hops. Through chains of paired friends, the human entity behind the identity is verified which ensures that the identity is not sybil. On the downside, secure pairing requires conscious user interaction and cannot be performed automatically. Thus, the resulting graph will only be loosely connected without guarantees of regular interactions.

The procedure works as follows. Each time a node is encountered the friends lists are exchanged and saved in a friendship graph G_F. The friendship graph is organized in L_d levels, comprising nodes at the same distance d from the local node n_0. Edges only exist between nodes in sequenced levels. The graph is constructed by a slightly modified breadth-first search (BFS) algorithm. The modification consists of allowing various edges from nodes in L_d to connect to the same single node in L_{d+1}. For every node in the friendship graph G_F, a trust value te_i is calculated according to Algorithm 1.

Algorithm 1 Explicit Social Trust

1: n_i: A node (local node: n_0)
2: $e_{i,j}$: Edge from n_i to n_j
3: FR_i: Set containing all friends of n_i
4: G_F: Friendship Graph of n_0
5: te_i: Explicit social trust value of n_i
6: L_d: Set of nodes with distance d from n_0 in G_F
7: $te_i = 1 \; \forall \; n_i \in L_1$
8: **for all** nodes n_i in proximity **do**
9: acquire FR_i from n_i and update G_F
10: build G_F and get $L_d \; \forall \; d$
11: **for all** $d \geq 1$ **do**
12: **for all** n_j in L_{d+1} **do**
13: $te_j = \sum_{n_k \in L_d : \exists e_{k,j}} \frac{te_k}{max(\sum_{n_l \in L_{d+1} : \exists e_{k,l}} 1, \; c) \cdot d}$
14: **end for**
15: **end for**
16: **end for**

The algorithm gives all direct friends a trust value of 1. A portion of each node's trust propagates to the next level, depending on the number of child nodes, resulting in each node receiving some trust from each parent node. This results in more trust for well connected nodes (i.e. with many parent nodes). Since the number of nodes increases with each level, a node's trust decreases with the hop distance d, depending on the connections to the previous level. To force this decrease in sparse graphs, e.g. chains, a minimum degradation factor c is introduced.

An example of how Algorithm 1 propagates trust is shown in Figure 1. The root of the graph is the local node followed by the direct friends on level L_1. The dashed lines are friend ties that exist, but are ignored by the algorithm, since the nodes are on the same level. In the circle are the trust values calculated by the root node with $c = 2$. The algorithm allows for $te > 1$ which is the case for one of the nodes on L_2. Since no node should be more trusted than a direct friend, te is bound to 1.

An evaluation of the performance, the expected trust distribution, and security concerns will be given in Section IV-A.

[1] According to Swamynathan et al. [24] transitivity is valid for up to 6 hops.

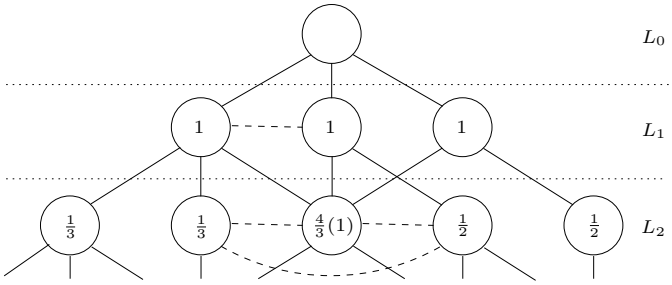


Fig. 1. Friendship Graph G_F

B. From Contacts to Implicit Social Trust

In everyday life, there are certain individuals we regularly share the same space or activity with, i.e. the familiars. These familiars can be easily identified by analyzing contact duration and/or contact frequency of the surrounding peers and sharing this information with those.

The advantage of this approach is the automatic operation without the need for conscious user interactions (e.g. pairing). Compared to the friendship graph, the mobility dynamics are captured, resulting in more opportunities to establish trusted relations in the vicinity. However, this approach cannot guarantee that a certain entity is behind the proclaimed identity and thus is not as secure as explicit social trust. Nevertheless, a certain amount of trust in a familiar can be justified since the identity cannot be a fast living, which is useful against sybil attacks.

Algorithm 2 Implicit Social Trust

1: n_i: A node (local node: n_0)
2: $f_{i,j}$: Familiarity value n_i has for n_j
3: F_i: Set containing $f_{i,j}$ of all n_j
4: ti_i: Implicit social trust value of n_i
5: $fs_i = \sum_j f_{i,j}$
6: **for all** nodes n_i in proximity **do**
7: update $f_{0,i}$
8: acquire F_i from n_i
9: **for all** n_j **do**
10: $ti_j = \underbrace{\frac{f_{0,j}}{fs_0}}_{\text{familiarity}} + \underbrace{\sum_k \frac{f_{0,k}}{fs_0} \cdot \frac{f_{k,j}}{fs_k - f_{k,0}}}_{\text{similarity}}$
11: **end for**
12: **end for**

Implicit social trust relies on the familiarity and the similarity of the nodes. Familiarity denotes the accumulated contact time and similarity describes to which degree two nodes familiars coincide. Both values are normalized, so the sum of all familiarities and all similarities is 1 each. Algorithm 2 keeps the familiarity values $f_{0,i}$ up to date by keeping track of the connection times with the surrounding nodes similar to regular community detection algorithms [6]. The set of all familiarity values is exchanged with all encountered nodes, thus giving a node a local approximation of the weighted network graph at 2 hops[2]. The implicit social trust ti_j in another node j is calculated by adding its familiarity and similarity (see Line 10 in Alg. 2). This results in a trust value in the range [0,2), whereas values greater than 1 are negligibly rare.

[2]Two hops are enough, since the purpose of implicit trust is to asses the surrounding nodes. For further hops, explicit trust or a reputation system has to be used.

	phone	protein	facebook	SW	CAVE	SF
# nodes	76	1846	63730	100	100	100
avg. degree	2.95	2.39	25.64	~4.00	~4.00	~3.80
diameter	9	19	14	~8.3	~11.5	~7.2
clustering coeff.	0.26	0.07	0.15	~0.26	~0.50	~0.05

TABLE I
RAW GRAPH PROPERTIES

C. Combining Metrics

The explicit and implicit social trust can be combined together to a consolidated social trust value. A trusted interaction partner is then identified if the consolidated trust is above a certain threshold th_{lu}, i.e.: $w_e \cdot t_e + w_i \cdot t_i \geq th_{lu}$ where w_e and w_i represent the weights for explicit and implicit trust, respectively. The weighting of both trust values may depend on the user and environment as discussed in Section V.

IV. EVALUATION

In this section we determine the complexity of establishing trust, analyze how trust propagates and discuss security related aspects, such as the resilience to sybils, using real-world graphs as well as synthetic graph models. The evaluation is done for both, the implicit and explicit social trust, whereas the evaluation of the combined social trust is omitted due to space constraints and left for future work

A. Explicit Social Trust

We rely on three real-world graphs consisting of the phone records of the MIT Reality traces [25], the protein interaction network [26] and the facebook graph from the New Orleans network[3] [27]. Some properties of the raw graphs are shown in Table I. Additionally, synthetic graphs based on the small-world (SW), caveman (CAVE) and scale-free (SF) model were used. For all graphs, edges represent friend ties (i.e., secure pairing). These graphs are processed to compute the friendship graph (G_F). The Watts and Strogatz model [28] is used to construct the small-world graph. The n nodes of the network are arranged to a ring and edges are established with k of their neighbors. Then, each edge is rewired with a probability p to a random node outside the k neighbors. The same is done for the caveman model, $(k+1)$-cliques are build and each edge is rewired to a node outside the clique with probability p. The caveman model differs from the small-world model by having non-overlapping communities. In the scale-free model, each node is assigned a popularity according to the power law distribution with minimum value s and shape α. Then, nodes are chosen independently at random and connected according to their popularity. The parameters that were used for the following evaluation, are $n = 100$, $k = 4$, $p = 0.01$, $s = 2$ and $\alpha = 2$ unless otherwise specified. All simulations results are the average over all possible combinations with multiple runs for the synthetic models. The graphs represent the steady-state regime of pairing i.e., after one has paired with most of his trusted friends and has encountered most of his possible contacts to acquire their list of paired friends (FR_i). We leave the analysis of the transient behavior with respect to mobility and pairing dynamics for future work.

Complexity: The calculation of the trust values te, can be performed during the construction of G_F by the modified BFS algorithm resulting in a complexity of $O(b^d)$ with b being

[3]Thanks to Alan Mislove for the data set.

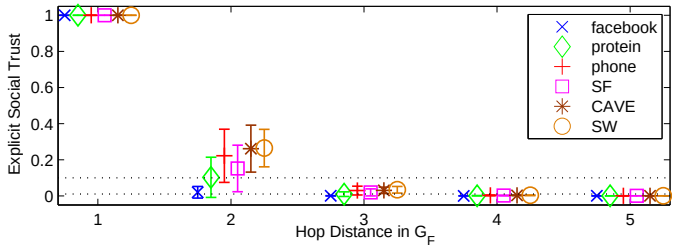


Fig. 2. Mean trust per node for each level in G_F

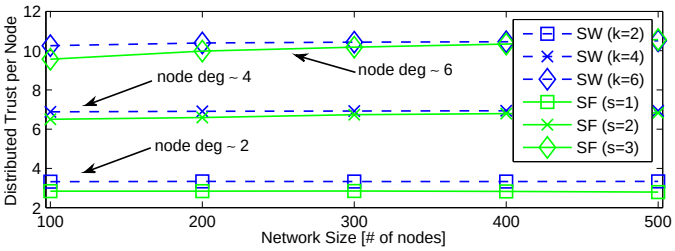


Fig. 3. Mean trust per node for increasing network size

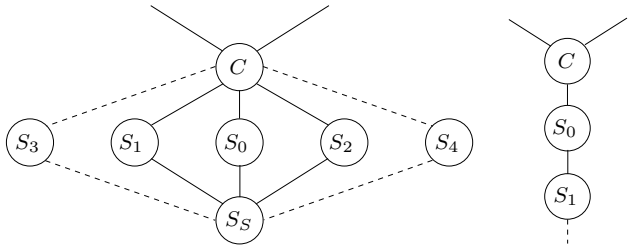
the branching factor (i.e. number of friends) and d being the depth of the resulting tree. To improve scalability for large n, we treat nodes with a trust value below 0.01 as leaves, thus ignoring all links they might have to lower levels of the tree. This reduces the average time for trust computation by 99% for the facebook graph.

Regarding the communication overhead, it depends only on the amount of friends a node has and thus scales with $O(b)$.

Trust Propagation: Algorithm 1 makes sure, each direct friend has a trust value of 1. Trust propagates through the direct friends to their friends on the next level and so on. One property of our algorithm is that the overall trust per level d, T_d is constant i.e., $\forall d,\ T_d = T_1 = |L_1|$. As the tree widens, trust is divided among more nodes and at some point the individual trust becomes negligible.

Figure 2 shows the average trust value (with std. dev.) assigned to nodes for a given level (or hop distance) for all graphs. For the calculation of te, c is set to 2 and nodes with $te < 0.01$ are treated as leaves, resulting in no nodes having trust past a distance of 5. The dotted lines mark the trust values of 0.1 and 0.01 as a reference. We can see that most of the nodes achieving a trust value higher than 0.1 are under 3 hops away. Furthermore, this trust value is a good tradeoff to capture a large portion of the network that can still be reasonably trusted. The trust values in the facebook graph degrade faster over the hop distance due to the graph's high average node degree (see Table I). A fast degradation with hop distance does not mean that the total amount of distributed trust ($\sum_d T_d$) is smaller as Table II(a) shows. Actually, the amount of trust received by a node mainly depends on its degree as shown next. Figure 3 shows the average total trust per node in G_F for different network sizes. The mean trust per node degree is around 1.6 for all the graphs. From both figures, we can conclude that the propagated trust does not depend on the network size nor its structure but on the average degree of a node. This makes our approach highly scalable.

Security: Since explicit trust is based on pairing, its resilience relies on the user's understanding of the necessity of only selecting trustworthy peers to pair with. Nevertheless, a device may be compromised with malware, an orthogonal problem which is beyond the scope of this paper. Detecting misbehavior of compromised nodes is the task of a reputation system and will not be discussed here. However, a compromised node, C, can be used as an anchor point for sybil nodes, S_x with $x \in \mathbb{N}$. This way, a compromised node may increase the influence over other nodes or outsource misbehaviors to sybils to remain undetected.



(a) Scenario 1 (b) Scenario 2

Fig. 4. Sybil Scenarios

To analyze the influence of generated sybil users of one compromised node, we consider two scenarios that differ in the strategy used to add sybils to the graph (see Figures 4(a) and 4(b)). In Scenario 1, all sybils are connected to the compromised node whereas in Scenario 2, a chain of sybils is built. Figure 5(a) shows for Scenario 1, the ratio of trust assigned to a sybil by all other nodes to trust a legitimate node gets on average. We compute this ratio (or percentage) as a function of the number of generated sybils under the compromised node. For all graphs, the trust in a sybil is never as high as in a normal user and decreases with the increasing number of sybils. It converges quickly to a low percentage and actually, considering a trust threshold of 0.1, only up to 10 sybils have an assigned trust value above it. The special sybil user S_S from Figure 4(a) is less dependent on the amount of sybils and stabilizes at around $15 - 20\%$ as can be seen in Figure 5(b). Similar results are obtained for Scenario 2 but are not shown due to space constraints. A more severe case would be to have cooperating compromised nodes, resulting in better connectivity for the generated sybils. This scenario is not evaluate since we assume that the infiltration of an actual node is hard.

Why Not Use Certificate Chained-Based Approaches: With PGP or Capkun et al. [18], trust is transitive independently of the chain length or the number of disjoint paths. A sybil user would thus only need to establish one trusted relation to gain full trust with all the others. Other approaches such as [7] do not allow for transitivity and only paired friends are trusted. Therefore sybils have to establish trust with all victims one by one. This conservative approach increases time and complexity resulting in very sparse trust relations.

Our approach is a tradeoff between those approaches that allows friendship transitivity depending on the hop distance and connectivity in the social graph. Trust in well-connected friends of friends will increase the number of trust relations but similar to Sybilguard [9], sybil users are ignored because of their low social interconnection.

B. Implicit Social Trust

To evaluate implicit social trust we use two real-world traces consisting of the MIT Reality traces [25] and the Haggle Infocom'05 traces [5]. Additionally, two synthetic

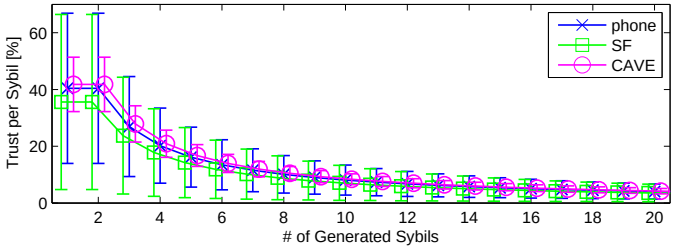
(a) Explicit Trust Value Statistics

	phone	protein	facebook	SW	CAVE	SF
Total distributed trust per node	4.61	3.91	46.93	6.87	5.80	6.50
# nodes w/ trust ≥ 0.10	8.2	6.4	50.4	12.5	9.5	12.3
# nodes w/ trust ≥ 0.01	16.4	22.5	515.8	29.75	20.4	37.9

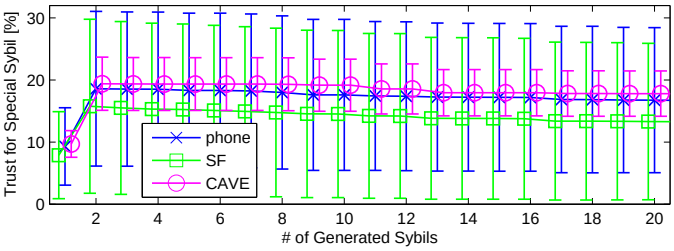
(b) Implicit Trust Value Statistics

	MIT	Haggle	SW	CAVE
Total distributed trust per node	2.00	2.00	2.00	2.00
# nodes w/ trust ≥ 0.10	5.4	2.5	6.7	8.6
# nodes w/ trust ≥ 0.01	26.65	37.3	30.7	22.49

TABLE II
TRUST VALUE STATISTICS



(a) Trust per Sybil



(b) Trust for Special Sybil

Fig. 5. Sybil Scenario 1

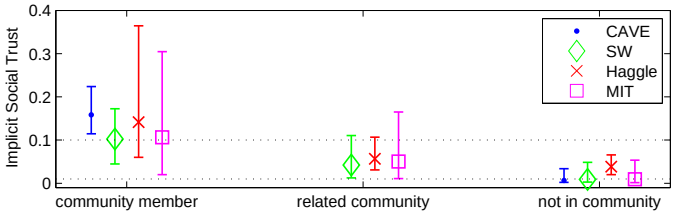


Fig. 6. Distribution of Trust Values in Local Community

mobility contact processes based on small world (SW) and caveman (CAVE) graphs were used. The underlying graphs are constructed as described in the previous section apart from k being set to 10. The contacts are simulated in the following way: a node n_i is chosen uniformly at random and its contact node n_j is chosen uniformly at random either from n_i's neighbors in the underlying graph with probability q or from all n nodes with probability $(1-q)$. The duration of a contact is power law distributed with minimum value s_d and shape α_d. For the following evaluation 30000 contacts are generated with $q = 0.5$ and the values $s_{dn} = 20$ and $\alpha_{dn} = 1$ for neighbors and $s_{dr} = 5$ and $\alpha_{dr} = 2$ for the random contacts.

Complexity: Algorithm 2 considers only one's familiars and their familiars and has thus complexity $O(b^2)$, b being the branching factor, i.e. the average number of familiars. Although the complexity is much lower than for the explicit social trust with $d > 2$, it can still be computationally intensive for a large number of familiars. In order to keep this aspect under control, an appropriate aging mechanism is necessary as discussed in the next section.

As far as data transfer is concerned, only the list of familiarity values has to be exchanged, thus the exchanged data is in the order of $O(b)$

Trust Propagation: The maximal propagated trust per node is bound to 2 as seen in Table II(b). Half of the assigned trust is based on the familiarity and the rest is based on the similarity of the nodes (see Alg. 2). Although the amount of trust is limited, the number of trusted nodes are still comparable to explicit social trust (compare Table II(a) and II(b)).

To understand how our algorithm behaves, we compared it against the modularity-based hierarchical community detection algorithm of Blondel et al. named Louvain [21]. Figure 6 represents the average (and std. dev.) values of implicit trust for all nodes assigned to the categories identified by the Louvain algorithm (i.e. community member, in related community, not in community). The dotted line marks the trust values of 0.1 and 0.01 as a reference. The correlation of trust values and the Louvain categories is evident, our algorithm can hence indirectly expose the structure of the local communities in the network. The caveman process has a clear community structure, thus lacking related communities and showing the best correlation. The communities detected in the real-world traces are large, resulting in large trust variances.

Security: The main goal of implicit social trust is to make sure the node is not a fast switching identity. The process of assigning trust should be resilient to attacks. Algorithm 2 assesses trust of a node by the node's familiarity and similarity. To become trusted with a target, with respect to normalized familiarity, an attacker would have to increase its familiarity and decrease the familiarity of all the target's familiars. For the former, the attacker needs to be physically near the target and for the latter, to jam all beacons in the targets surrounding which requires a big effort and is easily detectable. Likewise, the similarity can be forged by increasing the familiarity with all nodes in the targets familiar set, also requiring physical presence. Cooperation among the sybils may also improve their received trust by pretending high similarity. However, transitivity of trust is very limited and sybils have to be present to gain influence, hence the incorporation of additional sybils, as for the explicit social trust is not as effective but we will evaluate possible attack scenarios and enhance our approach in future work. Nevertheless, mobility anomaly detection as well as other sybil countermeasures [8], [10] can be used to further increase the effort needed for an attack.

Why Not Use Classical Community Detection: One may wonder why we did not use a community detection algorithm such as the one proposed by Hui et al. [6] in the first place since our approach achieves the same, indirectly. The reason is their discrete output and ease of manipulation especially for distributed versions. A community detection algorithm usually has a binary output, either a node is in one's community or not. Hierarchical algorithms (e.g. Louvain) may produce a non-binary output, but mapping the hierarchies to trust values is still inaccurate and not very meaningful. Since nodes at the

core of a community and at the border should not have the same trust values, especially in large communities, classical community detection is insufficient.

The bigger issue however, is the manipulation-proneness of distributed community detection algorithms. The three distributed algorithms *Simple*, *k-Clique* and *Modularity* proposed by Hui et al. can all be manipulated in several ways by an attacker in order for him/her to be included in a community by exchanging manipulated familiarity and community sets for example. With our approach, trust assigned to a node only depends on direct observables (i.e. contact time), without relaying on information received by that node (i.e. the nodes familiars set).

V. Discussion and Future Directions

One rationale behind our approach using explicit and implicit social trust is to have smooth levels of trust between untrusted and trusted relations as in real-world social networks. Transitivity in certification chains and community detection define no barriers or strict barriers among social categories, respectively, which may not reflect reality accurately.

We have, however, not discussed thoroughly how to weight explicit and implicit social trust. Depending on the environment (e.g., friendly vs. unfriendly), explicit and implicit social trust should be weighted dynamically, for example, by putting more weight in the former in unfriendly environments. But actually even explicit trust itself could output different trust values by not treating equally direct friend ties (and their successors in the graph) such as co-workers, schoolmates, friends or family members with whom we are paired; this to reflect different trust for different affairs. Also related to the environment is the representativeness of the implicit social graph as a user will evolve in different communities with time. An aging mechanism has to be applied to remove old or random encountered users and not end up with a clique resulting in a meaningless even structure. Preliminary results show that dynamic aging based on the online contact aggregation approach by Hossmann et al. [29] provides more robustness to the implicit social trust by searching for the optimal representation of the current underlying social network.

So far, we assumed users to be embedded in social environments with social trust (either explicit, implicit or both). But what if none of the surrounding peers is in either social graphs such as a user traveling alone in a new city? Then no interactions would be triggered. In this case, adapting aging to fast changing environments would help. As an alternative, reputation systems can provide higher levels of trust. Yet, more security can be brought to received ratings by weighting them with the social trust and hence counteract liars (sybils). Future work must investigate the interrelation between social trust and reputation systems.

For all these reasons, we will further investigate how we can leverage and adapt to the context and environment.

VI. Conclusion

In this paper, we have shown the importance of reconsidering the fundamental level of trust in opportunistic networks i.e., trust in an identity being genuine and honest as opposed to fake identities also known as sybils. We proposed two secure and scalable algorithms to assess explicit and implicit social trust. We have shown that approaches such as PGP-like chains and community detection are not suited for trust establishment in opportunistic networks. With our approach, the number of influential sybils in bounded e.g., to a maximum of 10 for SW graph ($k = 4$), independently on the network size. We believe that our approach has many fields of applicability from securing DTN routing/forwarding to more resilient reputation systems.

References

[1] G. Karlsson, V. Lenders, and M. May, "Delay-tolerant broadcasting," *IEEE Transactions on Broadcasting*, vol. 53, March 2007.

[2] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing robust and ubiquitous security support for mobile ad hoc networks," in *ICNP*, 2001.

[3] P. R. Zimmermann, *The Official PGP User's Guide*. MIT press, 1995.

[4] J. R. Douceur, "The sybil attack," in *IPTPS*, 2002.

[5] A. Chaintreau, P. Hui, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on opportunistic forwarding algorithms," *IEEE TMC*, vol. 6, 2007.

[6] P. Hui, E. Yoneki, S. Y. Chan, and J. Crowcroft, "Distributed community detection in delay tolerant networks," in *MobiArch*, 2007.

[7] S. Capkun, J.-P. Hubaux, and L. Buttyan, "Mobility helps peer-to-peer security," *IEEE TMC*, vol. 5, 2006.

[8] C. Piro, C. Shields, and B. N. Levine, "Detecting the sybil attack in mobile ad hoc networks," in *Securecomm and Workshops*, 2006.

[9] H. Yu, M. Kaminsky, P. B. Gibbons, and A. D. Flaxman, "Sybilguard: Defending against sybil attacks via social networks," *IEEE/ACM ToN*, vol. 16, 2008.

[10] A. Tangpong, G. Kesidis, H. yuan Hsu, and A. Hurson, "Robust Sybil Detection for MANETs," in *ICCCN*, 2009.

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

[12] D. Quercia, S. Hailes, and L. Capra, "B-trust: Bayesian trust framework for pervasive computing," in *iTrust*, 2006.

[13] J. Liu and V. Issarny, "Enhanced reputation mechanism for mobile ad hoc networks," in *iTrust*, 2004.

[14] K. Walsh and E. G. Sirer, "Experience with an object reputation system for peer-to-peer filesharing," in *NSDI*, 2006.

[15] D. Quercia, S. Hailes, and L. Capra, "Lightweight distributed trust propagation," in *ICDM*, 2007.

[16] D. Quercia, S. Hailes, and L. Capra, "MobiRate: Making Mobile Raters Stick to their Word," in *UbiComp*, 2008.

[17] R. Housley, W. Ford, W. Polk, and D. Solo, "Internet x.509 public key infrastructure certificate and crl profile," 1999.

[18] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Self-organized public-key management for mobile ad hoc network," *IEEE TMC*, vol. 2, 2003.

[19] Y. Lin, A. Studer, H. Hsiao, J. McCune, K. Wang, M. Krohn, P. Lin, A. Perrig, H. Sun, and B. Yang, "SPATE: Small-group PKI-less Authenticated Trust Establishment," in *MobiSys*, 2009.

[20] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Physical Review E*, vol. 69, 2004.

[21] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, 2008.

[22] A. Clauset, "Finding local community structure in networks," *Physical Review E*, vol. 72, 2005.

[23] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay tolerant networks," in *MobiHoc*, 2008.

[24] G. Swamynathan, C. Wilson, B. Boe, K. Almeroth, and B. Y. Zhao, "Do social networks improve e-commerce?: A study on social marketplaces," in *WOSN*, 2008.

[25] N. Eagle and A. (Sandy) Pentland, "Reality mining: sensing complex social systems," *Personal and Ubiquitous Computing*, vol. 10, 2006.

[26] H. Jeong, S. P. Mason, A. L. Barabasi, and Z. N. Oltvai, "Lethality and centrality in protein networks," *Nature*, vol. 411, 2001.

[27] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in facebook," in *WOSN*, 2009.

[28] D. J. Watts, *Small Worlds : The Dynamics of Networks between Order and Randomness*. Princeton University Press, 2003.

[29] T. Hossmann, T. Spyropoulos, and F. Legendre, "Know thy neighbor: Towards optimal mapping of contacts to social graphs for dtn routing," in *IEEE Infocom*, 2010.