Design and Verification of Trusted Collective Adaptive Systems

ALESSANDRO ALDINI, University of Urbino "Carlo Bo"

Collective adaptive systems (CAS) often adopt cooperative operating strategies in order to run distributed decision-making mechanisms. Sometimes, their effectiveness massively relies on the collaborative nature of individuals' behavior. Stimulating cooperation while preventing selfish and malicious behaviors is the main objective of trust and reputation models. These models are largely used in distributed, peer-to-peer environments and, therefore, represent an ideal framework for improving the robustness, as well as security, of CAS. In this paper, we propose a formal framework for modeling and verifying trusted CAS. From the modeling perspective, mobility, adaptiveness and trust-based interaction represent the main ingredients used to define a flexible and easy-to-use paradigm. Concerning analysis, formal automated techniques based on equivalence and model checking support the prediction of the CAS behavior and the verification of the underlying trust and reputation models, with the specific aim of estimating robustness with respect to the typical attacks conducted against webs of trust.

CCS Concepts: •**Security and privacy** → **Trust frameworks; Logic and verification;** •**Theory of computation** → **Process calculi; Verification by model checking;** •**Computing methodologies** → **Model verification and validation;**

Additional Key Words and Phrases: Collective adaptive systems, trust and reputation models, concurrency theory

1. INTRODUCTION

The collaborative nature of the behavior of the individuals forming complex, distributed systems represents a fundamental aspect opening up opportunities to improve the effectiveness and the efficiency of these systems [Saied et al. 2013]. In particular, collective adaptive systems (CAS) consist of individuals cooperating to achieve a community-based objective in a highly adaptive and open environment. The design and verification of CAS represents a challenge because of several orthogonal properties related to mobility, flexibility, adaptiveness, and openness. Hence, the assessment of their qualitative and quantitative behaviors requires various design approaches and verification techniques [Hillston et al. 2014; Bernardo et al. 2016]. Here, we concentrate on one of the aspects that received less attention in spite of its strong relation with the foundation principles of CAS, and that is the attitude to cooperation. In fact, as the need for cooperation becomes pervasive, more and more security threats come into play, which are related to the protection of the community not only from external attacks, but also with respect to insider threats. These may depend on selfish or malicious behaviors of agents, either in isolation or in collusion, which are due to, e.g., the competitive nature of service and resource allocation policies. Securing mechanisms *per se* do not offer protection against such behaviors, which must be mitigated through adequate incentive mechanisms.

Trust and reputation management systems [Jøsang 2007] stimulate the attitude to cooperation in several different domains, ranging from participatory sensing sys-

Author's address: A. Aldini, Department of Pure and Applied Sciences, University of Urbino "Carlo Bo", Piazza della Repubblica 13, 61029 Urbino, Italy.

DOI: http://dx.doi.org/10.1145/0000000.0000000

tems and wireless sensor networks to peer-to-peer and user-centric platforms. Trust represents the belief of an agent requiring a task about the capability and the intention of another agent of performing such a task. Its evaluation depends on direct observations, e.g., obtained through watchdog mechanisms monitoring the behavior of neighbors, and on recommendations received by other agents collaborating to maintain the reputation profile of every agent. As an example, trustworthy sensor networks base their ability to collectively process sensed data on decentralized reputation systems [Ganeriwal et al. 2008; Yu et al. 2012; Han et al. 2014; Tarable et al. 2015; Mousa et al. 2015]. In these frameworks, trust and reputation are distributed over all the network agents, which exchange and aggregate personal opinions, calculate trust scores of target agents, and disseminate such values [Packer et al. 2014; Yaich et al. 2012; Cho et al. 2011; Momani 2010; Buchegger and Boudec 2004].

Trust can be given a computational formalization with the aim of estimating the belief mentioned above either statistically or deterministically. Therefore, based on trust metrics and thresholds, collaboration among agents with high reputation can be strengthened while selfish and malicious agents can be excluded from the community, thus supporting activities like intrusion detection and participatory sensing. Webs of trust are usually established either on the base of a geographical notion of group of agents, as in crowdsourcing and sensor networks [Ganeriwal et al. 2008], or by following community based models, as in social networks and P2P environments [Zhang et al. 2007].

The need for formal verification of trust and reputation models is motivated by (i) the variety of aspects coming into play, including the flexibility and the dynamic behavior of the web of trust [Yu et al. 2012; Marmol and Perez 2009], and (ii) the complexity of the potential attacks and related countermeasures, which include [Marmol and Perez 2009]:

— bad mouthing: negative feedback reported by an adversary about the behavior of a trusted agent;
— ballot stuffing: positive feedback reported by an adversary about the behavior of a malicious agent;
— collusion: attack conducted by multiple adversaries which act together with the aim of damaging a honest agent;
— on-off: attack conducted by an adversary alternating between normal behaviors and misbehaviors.
— sybil: attack conducted by an adversary generating multiple identities with the aim of flooding the system with fake information or misbehaviors.
— white-washing: attack conducted by a misbehaving adversary who leaves the system whenever her reputation is compromised and then rejoins it using a different identity.

In this paper, we introduce a uniform, formal framework encompassing different analysis techniques supporting the systematic verification of trusted CAS with respect to the above attack taxonomy. The proposed framework is based on a process algebraic specification language for the description of CAS-like systems. Then, trust model and system model are specified separately and integrated transparently at the semantics level. For analysis purposes, we apply both equivalence and model checking techniques.

The main, novel contributions of such an approach can be summarized as follows:

— a (trust-based) communication model favoring flexibility, usability, and scalability of the system specifications, where trust-based policies are inspired by the noninterference approach to information flow analysis;

— a parametric modeling framework separating completely the specification of the system behavior from the description of the trust model, with the aim of favoring model (re)-usability;
— a unifying framework favoring the analysis of the behavior of CAS embedding trust models of communication;
— equivalence checking as a tool for the qualitative comparison between different trust models and for the verification of the robustness against the various classes of attacks.

In [Aldini 2015], an analogous formal approach is proposed to model and analyze trust-based concurrent systems. With respect to the current paper, it is based on a specific, less flexible trust infrastructure that is not designed to support distributed, dynamic, adaptive systems. Moreover, the trust based policies are not based on the noninterference approach to information flow analysis. From the analysis standpoint, it considers only model checking techniques for quantitative verification based on trust information. The current paper extends previous work [Aldini 2016], mainly by integrating it with analysis techniques, based on both equivalence checking and model checking. In order to demonstrate the capabilities of such a verification framework, we present the analysis of two real-world case studies.

The paper is organized as follows. In Section 2, we discuss some of the characteristics of distributed trust and reputation models, which motivate the modeling solutions proposed in Section 3, where the process algebraic specification language is presented. In the same section, we discuss also the definition and application of a behavioral equivalence, equivalence checking techniques, a trust-based logic, and model checking techniques. In Section 4, we present the first case study: the analysis of the Trust-Incentive Service Management system [Zhang et al. 2007]. The objective is to show the capabilities of the modeling framework and how to estimate the robustness of the trust model through sensitive analysis based on equivalence and model checking. In Section 5, we show the application of our unifying framework to the comparison of two different trust models, the Reputation-based Framework for Sensor Networks [Ganeriwal et al. 2008] (RFSN) and the Robust Reputation System [Buchegger and Boudec 2004] (RRS). The automatic verification of both case studies has been possible through a mapping from our formal framework to NuSMV [Cimatti et al. 2002]. Some conclusions and comparison with related work terminate the paper in Section 6.

2. TRUST IN COLLABORATIVE ADAPTIVE SYSTEMS

In the literature, computational notions of trust and trust management issues have long been investigated in mobile ad-hoc networks (MANETs) and wireless sensor networks (WSNs).

The lack of centralized systems collecting and sharing trust values, which is typical of distributed systems, emphasizes the role of reputation sharing mechanisms. EigenTrust [Kamvar et al. 2003] is a popular trust system originally proposed for P2P file sharing systems aiming at approximating in a purely distributed framework a global, centralized notion of reputation. Agents assign value 1 (resp., -1) to each satisfactory (resp., unsatisfactory) interaction. Then, the basic version of the algorithm works as follows. To compute the local trust of I about J, called s_{IJ}, all the scores assigned by I to interactions with J are summed up. Local trust values of I about its neighborhood are then normalized with respect to $\sum_K s_{IK}$, as follows:

$$c_{IJ} = \frac{\max(s_{IJ}, 0)}{\sum_K \max(s_{IK}, 0)}$$

to ensure that all values will be between 0 and 1 and that $\sum_J c_{IJ} = 1$. These trust values are then aggregated to form a distributed notion of reputation. The principle behind the computation of the trust t_{IJ} from I to J is to combine the opinions of I's neighbors, as follows:

$$t_{IJ} = \sum_K c_{IK} c_{KJ}$$

In matrix notation, given C the matrix $[c_{IJ}]$ of all the trust values, C^T its transposition, and c_I the vector containing the values c_{IJ}, then the vector t_I of the values t_{IJ} is computed as $C^T \cdot c_I$. Such a mechanism can be iterated by aggregating the opinions of communities in cascade, i.e., by computing $(C^T)^n \cdot c_I$. For n large enough, the result converges to the same trust vector for every agent I in the network, which thus can be interpreted as a vector of global trust values.

In the setting of reputation-based sensor networks [Ganeriwal et al. 2008; Li et al. 2008], the local, direct trust from agent I to agent J is maintained by using a watchdog mechanism in I reporting the result of each direct experience with J. Such a feedback, which may consist of scores or, more simply, the amount of good behaviors and of misbehaviors observed, is then used to parameterize a trust metric relying on a standard Bayesian approach. The calculated trust value thus represents the expectation estimating the belief level that one agent has on another agent for a specific action. Second hand information can be provided by neighbor agents, in the form of recommended trust values that are scaled by a factor proportional to the trust towards such recommending agents. For instance, several models [Beth et al. 1994; Ngai and Lyu 2004; Zhang et al. 2007] propose the computation of the trust value of agent I towards agent J through agent K via a formula of the form $1 - (1 - t_{KJ})^{t_{IK}}$. Hence, t_{KJ} plays the role of a recommendation given to I, which is weighted by the direct trust from I to K. More complex policies can be used to combine direct experience, recommendations, and further parameters. As an example, in PeerTrust [Xiong and Liu 2004], trust towards an agent I depends on the amount of known interactions between I and other agents, the known feedback reported by such agents, the credibility of such agents, and an adaptive community context factor for agent I. In turn, credibility of an agent J from the viewpoint of an agent K depends on the recommendations about J provided by agents that previously interacted with both K and J.

In all these examples, the trust-based selection policy is based on the rule $t_{IJ} \geq th_I$, where the trust threshold th_I may depend on several factors influencing I, such as the dispositional trust of I, which represents the initial willingness of agent I to cooperate with unknown agents.

3. PROCESS ALGEBRA FOR WEB OF TRUST

All the examples shown in the previous section emphasize that the ingredients needed to feed a trust model for distributed, adaptive systems are:

(1) the set of direct experiences affecting a local notion of trust, that is, subjective from the viewpoint of an agent expressing quantitatively (through a positive/negative score) the quality of a direct interaction.
(2) the sets of agents (which we call groups) collaborating, e.g., through the exchange of recommendations, in order to share reputation information. It is worth observing that the composition of such groups may be characterized by high levels of dynamicity.

In the following, we abstract from the way in which the basic parameters concerned with trust and reputation are combined to compute metrics governing the decision making process, which is a task specific to the trust model. Instead, we concentrate on

the specification of the behavior of agents and on the establishment of their networks of trust. For this purpose, as we will see, in the semantics of our formal specification language we have rules describing (i) how the basic parameters needed by the trust model are calculated and maintained, and (ii) how the results computed by the trust model, i.e., the t_{IJ} values, are then used to govern the trust-based interactions. All the machinery taking in input the basic parameters mentioned above and returning as output the trust values is hidden and left to the specification of the trust model.

Moreover, to simplify the presentation, unless differently specified we restrict our consideration to systems in which one type of service is provided within the network.

3.1. Calculus for Sequential Processes

We start by presenting a basic calculus (see, e.g., [Fokkink 2007]) for the description of sequential processes. Let $Name$ be the set of action names, ranged over by $a, b, \ldots$, such that it is defined as the union of four disjoint sets, $\{\tau\} \cup Name_o \cup Name_i \cup Name_t$, where τ is the internal, invisible action, $Name_o$ and $Name_i$ represent the sets of output actions and input actions, respectively, and $Name_t$ denotes a set of special actions managing the establishment of a web of trust. The set of process terms of the basic calculus for sequential processes is generated through the following syntax:

$$P \ ::= \ \underline{0} \mid a \,.\, P \mid P + P \mid B$$

where we have the constant $\underline{0}$ for the inactive process, the classical algebraic operators for prefix and nondeterministic choice, and a constant based mechanism for expressing recursive processes. As usual, we consider only guarded and closed process terms. The semantics of process terms is expressed in terms of labeled transition systems.

Definition 3.1. A labeled transition system (LTS) is a tuple (Q, q_0, L, R), where Q is a finite set of states (with q_0 the initial one), L is a finite set of labels, and $R \subseteq Q \times L \times Q$ is a finitely-branching transition relation.

For notational convenience, $(q, l, q') \in R$ is denoted by $q \xrightarrow{l} q'$. Then, the behavior of process term P, denoted by $[\![P]\!]$, is defined by the smallest LTS $(Q, q_0, Name, R)$ such that Q is the set of process terms of our basic calculus (with P representing the initial state q_0), and the transitions in R are obtained through the application of the operational semantics rules of Table I.

In the following, we call *agent* any instance of a given process term. In other words, an agent represents an element exhibiting the behavior associated with a process term. The kernel of the semantics of an agent named I and belonging to the behavioral pattern defined by process term P is obtained from P by replacing each action a of P with $I.a$. Then, the semantics $[\![I]\!]$ of agent I derives from $[\![P]\!]$ in the same way. In essence, a process term P represents a behavioral type, while an agent I of such a type, denoted $I : P$, is an instance of P. Analogously, the notation $I.B$ expresses that the behavior of I is given by the process term identified by the constant B. Such a separation of concerns between the definition of agents (and the topology they form) and of their behavioral pattern is inspired by process algebraic architectural description languages (see, e.g., [Aldini et al. 2010]), and is motivated by usability and scalability issues.

3.2. Calculus for Interacting Processes

In the setting of concurrency, we deal with trust adaptive systems made of several interacting agents that obey a trust-based synchronous communication model, the rules of which depend, as motivated above, on (dynamic) group membership and (dynamic) trust relationships. Syntactically, we avoid the adoption of an explicit parallel composition operator, which would make it rigid and complicate the modelling of systems

Table I. Semantics rules of the basic calculus

prefix	$a \,.\, P \xrightarrow{a} P$	
choice	$\dfrac{P_1 \xrightarrow{a} P_1'}{P_1 + P_2 \xrightarrow{a} P_1'}$	$\dfrac{P_2 \xrightarrow{a} P_2'}{P_1 + P_2 \xrightarrow{a} P_2'}$
recursion	$B \stackrel{\text{def}}{=} P \quad \dfrac{P \xrightarrow{a} P'}{B \xrightarrow{a} P'}$	

consisting of a large number of agents where the topology of the interacting groups may change dynamically. Semantically, trust model and behavioral model are maintained separately, where the latter turns out to be a special kind of trust-based LTS, the states of which are enriched with the additional elements surveyed above, which we now introduce formally as follows.

Firstly, we use a notion of global state of the trust adaptive system, which is made of the local states of the agents involved. Formally, $\mathcal{S}$ is a finite set of agents $\{I_i : P_i \mid 1 \leq i \leq n\}$ such that each agent name I_i is unique. In the following, we employ $P, P', Q, Q' \ldots$ to represent the kernel of the semantics of agents, so that $P \xrightarrow{I.a} P'$ denotes a transition performed by agent I from the local state described by process term P to the local state described by process term P'. Given a set $\mathcal{S}$ of interacting agents, the vector of process terms expressing the local state of each agent in $\mathcal{S}$ represents a global state of the system, ranged over by $\mathcal{P}, \mathcal{P}', \ldots$. As a shorthand, $\mathcal{P}[P'/P]$ represents the substitution of P with P' in $\mathcal{P}$. Such a notation is not ambiguous as P, P' express the kernel of the semantics of a uniquely identified agent in $\mathcal{S}$.

Secondly, synchronous communication is based on the definition of the synchronization set $S \subseteq Name_o \times Name_i$, containing pairs of actions denoted syntactically by $a \times b$. Action a represents the output, governing counterpart of the synchronous communication, while action b denotes the input, reacting counterpart. Hence, we assume that synchronous communication is asymmetric, in the sense that one of the two agents involved governs it while the other one reacts. As we will see, such a mechanism is generalized to model broadcast communications, in which case we use the syntactic notation $a \otimes b$.

Thirdly, synchronous communication is allowed only within groups of agents. More precisely, given a set of sets of agents $\mathcal{G} \subseteq 2^{\mathcal{S}}$, each element of $\mathcal{G}$ represents a group of agents that can (i) communicate directly with each other, and (ii) share trust opinions. As we will see, group membership is managed dynamically through ad-hoc actions.

Fourthly, trust relationships are defined through the adoption of a trust model, which is fed by trust opinions deriving from personal observations. Opinions are provided through specific actions and are collected by the multiset of trust opinions $\mathcal{E}$, with support set of type $(\mathcal{S}, \mathsf{T} \cup \{?\})_{\mathcal{S}}$, where T is the trust domain[1]. Element $(J, ?)_I$ means that I is expected to evaluate the last direct interaction conducted with J. Element $(J, v)_I$ means that I has evaluated an interaction with J by assigning the score v to it. The assignment of a score v by I to an interaction with J is completely left to the subjective perception of agent I, which is a fundamental condition of trust systems. For this reason, we do not employ any predefined mapping from input/output actions to values in T and we do not make any assumption on the accuracy and correctness of the opinion[2]. We also point out that $\mathcal{E}$ is a multiset, as agent I may evaluate many behaviors

[1]Generally speaking, in real systems trust domains are, at least, partially ordered sets. For the sake of simplicity of the presentation, here we assume T to be totally ordered.

[2]Notice also that I may be either a honest agent rating only direct observations, or a malicious agent injecting fake opinions that are not motivated by real observations.

of agent J after several different direct interactions, some of which could be assigned the same score. Trust opinions have to be interpreted as local to the agent generating them, even if no assumptions are made about the way in which they are maintained, used, made available, and shared, which represent aspects depending completely on the specific trust model.

The last ingredient to consider is the trust model $\mathcal{T}$, which encompasses (i) a machinery taking in input the trust opinions and returning the trust values t_{IJ}, and (ii) the set of thresholds th_I, $I \in \mathcal{S}$, used by the agents to govern trust-based choices.

Example 3.2. In order to illustrate the formal functioning of the machinery behind the trust model, let us consider the encoding of EigenTrust [Kamvar et al. 2003] in our framework. First, we define the trust domain $\mathsf{T} = \{-1, 1\}$, because in EigenTrust scores are either positive (1) or negative (-1). Then, it is sufficient to observe that the local trust s_{IJ} from I to J is computed as follows:

$$s_{IJ} = mul((J, 1)_I) - mul((J, -1)_I)$$

where $mul(e)$ denotes the multiplicity of term e in $\mathcal{E}$. The trust values c_{IJ} and, therefore, t_{IJ} are obtained by applying the same operations defined in [Kamvar et al. 2003].

Based on the elements introduced above, the semantics of a trust adaptive system modeling the behavior of a set $\mathcal{S}$ of interacting agents is given by an extension of LTSs, called trust LTSs (TLTSs), where each state in Q is a quadruple of the form $(\mathcal{P}, S, \mathcal{G}, \mathcal{E})$, which represents: the global state $\mathcal{P}$ denoting the local behavior of each agent in $\mathcal{S}$, the synchronization set S governing the synchronous communication between agents, the dynamic set of groups $\mathcal{G}$ specifying the interacting communities to which the agents belong, and the dynamic multiset of trust opinions $\mathcal{E}$ deriving from such interactions.

Formally, let $\mathcal{T}$ be a trust model and the tuple $A = (\mathcal{S}, S, \mathcal{G}, \mathcal{E})$ represent a trust adaptive system, where $\mathcal{S} = \{I_i : P_i \mid 1 \leq i \leq n\}$, S is the synchronization set, $\mathcal{G}$ is the initial set of groups, $\mathcal{E}$ is the initial multiset of opinions. Then, the behavior of A under $\mathcal{T}$ is the smallest TLTS (Q, q_0, L, R) satisfying the following conditions:

— $q_0 = (\mathcal{P}, S, \mathcal{G}, \mathcal{E})$ is such that the n-length vector $\mathcal{P}$ of process terms represents the initial local state P_i of each agent I_i, $1 \leq i \leq n$;
— the transitions in R derive from the application of the operational semantics rules of Table II whenever the trust model is $\mathcal{T}$;
— the set of labels L, ranged over by α, contains τ and the names of the actions expressing interactions among agents (see the transitions in the conclusions of the operational semantics rules of Table II).

As far as the trust opinions are concerned, usually, $\mathcal{E} = \emptyset$ in q_0. However, in some case (see, e.g., [Kamvar et al. 2003]), a priori estimations of trust are assigned to agents that are known to be trustworthy in a community, e.g., as they are among the founders of the community. Pre-trusted agents can be considered by setting $\mathcal{E}$ appropriately in the initial state.

3.3. Semantics Rules of Interacting Agents

As anticipated above, the evolution of a trust adaptive system is determined through the application of the semantics rules of Table II, which formalize the parallel composition of the agents forming the system. Let us explain them intuitively.

Rule (int) refers to the action τ, which is performed autonomously by each agent.

Rules $(gr1)$ and $(gr2)$ describe autonomous, internal activities performed by each agent to manage their membership to communities. In particular, action $ent(G) \in Name_t$, where $G \in \mathcal{G}$, allows an agent to join the group G of agents. Notice that G is replaced by $G \cup \{I\}$, where I is the agent joining the group. Action $esc(G) \in Name_t$,

Table II. Semantics rules for parallel composition

$$(int)\quad \frac{P \in \mathcal{P} \quad P \xrightarrow{I.\tau} P'}{(\mathcal{P},S,\mathcal{G},\mathcal{E}) \xrightarrow{\tau} (\mathcal{P}[P'/P],S,\mathcal{G},\mathcal{E})}$$

$$(gr1)\quad \frac{P \in \mathcal{P} \quad G \in \mathcal{G} \quad P \xrightarrow{I.ent(G)} P'}{(\mathcal{P},S,\mathcal{G},\mathcal{E}) \xrightarrow{\tau} (\mathcal{P}[P'/P],S,\mathcal{G}[(G \cup \{I\})/G],\mathcal{E})}$$

$$(gr2)\quad \frac{P \in \mathcal{P} \quad G \in \mathcal{G} \wedge I \in G \quad P \xrightarrow{I.esc(G)} P'}{(\mathcal{P},S,\mathcal{G},\mathcal{E}) \xrightarrow{\tau} (\mathcal{P}[P'/P],S,\mathcal{G}[(G\backslash\{I\})/G],\mathcal{E})}$$

$$(tc1)\quad \frac{P,Q \in \mathcal{P} \quad a \times b \in S \quad G \in \mathcal{G} \wedge I,J \in G, I \neq J \quad P \xrightarrow{I.a} P' \quad Q \xrightarrow{J.b} Q' \quad a \in H \wedge t_{IJ} \geq th_I}{(\mathcal{P},S,\mathcal{G},\mathcal{E}) \xrightarrow{I.a \times J.b} (\mathcal{P}[P'/P,Q'/Q],S,\mathcal{G},\mathcal{E} \cup \{\!|(J,?)_I|\!\} \cup \{\!|(I,?)_J|\!\})}$$

$$(tc2)\quad \frac{P,Q \in \mathcal{P} \quad a \times b \in S \quad G \in \mathcal{G} \wedge I,J \in G, I \neq J \quad P \xrightarrow{I.a} P' \quad Q \xrightarrow{J.b} Q' \quad a \in L \wedge t_{IJ} < th_I}{(\mathcal{P},S,\mathcal{G},\mathcal{E}) \xrightarrow{I.a \times J.b} (\mathcal{P}[P'/P,Q'/Q],S,\mathcal{G},\mathcal{E} \cup \{\!|(J,?)_I|\!\} \cup \{\!|(I,?)_J|\!\})}$$

$$(tc3)\quad \frac{P,Q \in \mathcal{P} \quad a \times b \in S \quad G \in \mathcal{G} \wedge I,J \in G, I \neq J \quad P \xrightarrow{I.a} P' \quad Q \xrightarrow{J.b} Q' \quad a \notin \{H \cup L\}}{(\mathcal{P},S,\mathcal{G},\mathcal{E}) \xrightarrow{I.a \times J.b} (\mathcal{P}[P'/P,Q'/Q],S,\mathcal{G},\mathcal{E})}$$

$$(bro)\quad \frac{P,P_1,\ldots,P_n \in \mathcal{P} \quad a \otimes b \in S \quad \{I,I_1,\ldots,I_n\} \in \mathcal{G} \quad P \xrightarrow{I.a} P' \quad P_i \xrightarrow{I_i.b} P_i' \quad 1 \leq i \leq n}{(\mathcal{P},S,\mathcal{G},\mathcal{E}) \xrightarrow{I.a \otimes \{I_1,\ldots,I_n\}.b} (\mathcal{P}[P'/P,P_1'/P_1,\ldots,P_n'/P_n],S,\mathcal{G},\mathcal{E} \cup \{\!|(I,?)_{I_1}|\!\} \cup \ldots \cup \{\!|(I,?)_{I_n}|\!\})}$$

$$(op1)\quad \frac{P \in \mathcal{P} \quad G \in \mathcal{G} \wedge I,J \in G \quad (J,?)_I \in \mathcal{E} \quad P \xrightarrow{I.obs(v)} P'}{(\mathcal{P},S,\mathcal{G},\mathcal{E}) \xrightarrow{\tau} (\mathcal{P}[P'/P],S,\mathcal{G},(\mathcal{E}\backslash\{\!|(J,?)_I|\!\}) \uplus \{\!|(J,v)_I|\!\})}$$

$$(op2)\quad \frac{P \in \mathcal{P} \quad G \in \mathcal{G} \wedge I,J \in G \quad P \xrightarrow{I.obs(J,v)} P'}{(\mathcal{P},S,\mathcal{G},\mathcal{E}) \xrightarrow{\tau} (\mathcal{P}[P'/P],S,\mathcal{G},\mathcal{E} \uplus \{\!|(J,v)_I|\!\})}$$

where $G \in \mathcal{G}$, allows an agent to leave the group G of agents. Notice that G is replaced by $G\backslash\{I\}$, where I is the agent leaving the group. Both actions are autonomously executed by the agents and give rise to internal actions τ.

The following three rules describe the synchronous communication between two different agents. First, based on the communication model previously described, an interaction from I, offering output a, to J, reacting with input b, is possible if two conditions hold: $a \times b$ belongs to the synchronization set S, and there exists a group G of which both I and J are members. Therefore, groups are used to dynamically confine the sets of agents that can interact directly through synchronous communication. Groups are also used to define, in a given instant of time, the community referenced by an agent in order to share and obtain trust-based information.

Depending on the nature of the interaction, the communication from I to J may depend on the trust of I towards J. To this aim, inspired by the noninterference approach to information flow analysis [Goguen and Meseguer 1982], all the actions involved in trust-based communications are classified into two disjoint sets, H and L, denoting high-level and low-level actions, respectively, such that $(H \cup L) \subseteq Name$ and for each $a \times b \in S$ it holds that $a \in H$ if and only if $b \in H$ and $a \in L$ if and only if $b \in L$.

Rule $(tc1)$ refers to the case in which agent I offers output $a \in H$. In such a case, the communication is possible only if the counterpart J satisfies the trust-based selection policy based on the trust threshold th_I, i.e., $t_{IJ} \geq th_I$. Since the communication model is asymmetric, then the trust-based condition is applied only by the agent offering the output action, which governs the interaction. An example of high-level action is the

service request sent by an agent I to a trusted agent J. The same request would not be sent to an untrusted partner.

Rule $(tc2)$ refers to the case in which agent I offers output $a \in L$. In such a case, an interaction through a is enabled when the trust-based selection policy based on the trust threshold th_I is not satisfied by the counterpart J, i.e., $t_{IJ} < th_I$. The reason for having also low-level actions is that I may send some message to an untrusted partner J, like, e.g., a denial of service communication in response to a service request previously sent by J to I.

We emphasize that the trust-based selection policy is based on the metric t_{IJ}, i.e., the trust of I towards J as estimated by the trust model $\mathcal{T}$, which in turn employs the set of opinions collected during the system execution. Hence, the details of the computation of t_{IJ} do not affect the definition of the semantics for interacting processes.

In the conclusions of rules $(tc1)$ and $(tc2)$, terms $(J,?)_I$ and $(I,?)_J$ are added to the set $\mathcal{E}$ of local opinions. They serve as a placeholder stating that a feedback, in the form of a score $v \in \mathsf{T}$, could be provided by each of the two parties to evaluate the level of satisfaction in the interaction with the other party.

Rule $(tc3)$ refers to the case of interactions that are not related to trust, i.e., $a \notin \{H \cup L\}$. In such a case, the interaction involving a does not rely on trust-based requirements.

Rule (bro) refers to a broadcast communication from I to its neighbors. Syntactically, this is distinguished through the notation $a \otimes b \in S$, where a denotes the output offered by I and b represents the input counterpart for all the agents involved. Such agents may then be interested in estimating such an observation in terms of trust towards I. An example of (trusted) broadcast communication is given by the sharing of local data by a sensor node in the setting of participatory sensing. Such a node is expected to distribute sensed values to the neighborhood and to be evaluated in terms of trust for such an activity.

Rule $(op1)$ expresses the expected behavior of an agent evaluating an interaction previously conducted. To report such a feedback, we use the special autonomous action $obs(v) \in Name_t$, where $v \in \mathsf{T}$. Notice that the effect of such an action is to replace the symbol $?$ in $(J,?)_I$ with the score v. As we will see, managing automatically the relation between interaction and related feedback at the semantics level is useful to achieve usability and scalability results from the modeling standpoint. Moreover, the decoupling of the observation of an interaction from its subsequent evaluation increases the flexibility of the model as ensures high degrees of freedom about time, enabling conditions, and results of the trust estimation.

Sometimes, trust opinions are reported by agents even in the absence of direct interactions, e.g., by the effect of indirect observations through watchdog mechanisms or, simply put, as the consequence of malicious behaviors intended to create false recommendations (see, e.g., the attacks discussed in Section 2). Rule $(op2)$ manages such situations. Notice that, differently to rule $(op1)$, it is necessary to specify the agent that is subject to the trust feedback.

Well-formedness. As specified, trust opinions refer to the evaluation of interactions. Formally, the placeholder $(J,?)_I$ is added to the multiset $\mathcal{E}$ through the union operator $\cup$ [3]. Since it can occur in $\mathcal{E}$ with multiplicity 1 at most, we assume that a score assigned by I to J refers to the last interaction among them, so that if other, older, unevaluated interactions among them exist, they lose the possibility to be evaluated. Such an interpretation, which avoids complex ST-semantics like ap-

[3]Multiset union is defined as the multiset such that each element has the maximal multiplicity it has in either multisets.

proaches [van Glabbeek and Vaandrager 1987], is of practical interest as it allows to model easily the situation in which no feedback is reported, e.g., because the agent is neither stimulated nor interested to provide evaluations. Whenever the placeholder $(J,?)_I$ is removed, an element of the form $(J,v)_I$ is added to $\mathcal{E}$ through the multiset sum operator $\uplus$ [4], meaning that such an element may occur in $\mathcal{E}$ with multiplicity greater than 1. Notice that, if two different placeholders $(J,?)_I$ and $(J',?)_I$ occur in $\mathcal{E}$, then the execution of transition $I.obs(v)$ assigns score v either to J or to J', nondeterministically. Such a situation is avoided if the feedback is reported before the execution of a new interaction with another agent, as typical in trust-based systems, in which case we say that the system is well-defined.

Aging mechanism. Sometimes, trust models propose a periodic refresh of trust and reputation metrics as a way to stimulate continuity of cooperative behaviors. Typically, this is ensured by scaling trust opinions by an aging factor $w \in [0,1]$, which is applied according to two possible semantics, i.e., either whenever a new observation is added or simply as time passes. In the former case, we use a version of rules $(op1)$ and $(op2)$ which, in their conclusions, replace each existing term $(J,v)_I \in \mathcal{E}$ with $(J, w \cdot v)_I$ (notice that here I and J are exactly the agents mentioned in the premises of the rules). In the latter case, we apply the same substitution for each trust opinion in $\mathcal{E}$ in the conclusions of all the semantics rules. The specific rules to use are therefore determined by the trust model.

3.4. Behavioral Equivalence and Equivalence Checking

We now investigate a behavioral equivalence for networks of trust, by applying in our framework a classical notion of bisimulation [Gorrieri and Versari 2015]. As it can be easily adapted to both LTSs and TLTSs, here we propose the following definition of weak bisimulation.

Let $\Longrightarrow$ denote the reflexive and transitive closure of $\xrightarrow{\tau}$, and $\stackrel{\hat{a}}{\Longrightarrow}$ stand for $\Longrightarrow$ if a is internal and for $\Longrightarrow \xrightarrow{a} \Longrightarrow$ (simply denoted by $\stackrel{a}{\Longrightarrow}$) otherwise.

Definition 3.3. Let (Q, q_0, L, R) be a LTS. A relation $\mathcal{B}$ over Q is a weak bisimulation iff, whenever $(q_1,q_2) \in \mathcal{B}$, then for each $a \in L$ it holds that:

— if $q_1 \xrightarrow{a} q_1'$ then $q_2 \stackrel{\hat{a}}{\Longrightarrow} q_2'$ and $(q_1', q_2') \in \mathcal{B}$;
— if $q_2 \xrightarrow{a} q_2'$ then $q_1 \stackrel{\hat{a}}{\Longrightarrow} q_1'$ and $(q_1', q_2') \in \mathcal{B}$.

States q_1 and q_2 are weakly bisimilar if (q_1,q_2) is contained in some weak bisimulation. As usual, we use the notation $q_1 \approx q_2$ (weak bisimilarity) to denote the largest weak bisimulation relating q_1 and q_2. Moreover, if q_1 and q_2 are the initial states of the process terms P_1 and P_2, respectively, we also write $P_1 \approx P_2$. The relation $\approx$ is reflexive, symmetric, and transitive.

We recall that when composing agents to define a trust adaptive system, all the actions modeling group membership and trust operations become internal from the viewpoint of the system. Therefore, if we apply the weak bisimulation semantics to TLTSs as it is, the effect is that the equivalence relation abstracts away from such activities. Along the line of separation of concerns adopted so far, weak bisimulation abstracts also from the trust model and the parameters $\mathcal{G}$ and $\mathcal{E}$ of the trust adaptive system. Hence, it concentrates on the observable behavior of the system and on the

[4] Multiset sum is defined as the multiset such that each element has the sum of the multiplicities it has in both multisets.

effects on such a behavior of the trust model and of the parameters above. For these reasons, we use $\approx$ as given in Def. 3.3 also to relate TLTSs.

Then, we say that TLTS M with initial state q_0 and TLTS M' with initial state q'_0 are behaviorally equivalent, denoted $M \approx M'$, if $q_0 \approx q'_0$. The same notation extends to the systems originating such TLTSs. Hence, a trust adaptive system A under the trust model $\mathcal{T}$ and a trust adaptive system A' under the trust model $\mathcal{T}'$ are behaviorally equivalent if so are the corresponding TLTSs. Moreover, we have the following compositionality result with respect to the parallel composition of interacting agents.

THEOREM 3.4. *Let* $A = (\{I_1 : P_1, \ldots, I_n : P_n\}, S, \mathcal{G}, \mathcal{E})$ *and* $A' = (\{I_1 : P'_1, \ldots, I_n : P'_n\}, S, \mathcal{G}, \mathcal{E})$ *be two trust adaptive systems under the trust model* $\mathcal{T}$*. If* $P_i \approx P'_i$ *for* $1 \leq i \leq n$*, then* $A \approx A'$ *under* $\mathcal{T}$*.*

The proof of the theorem derives from two facts. First, the comparison of the two systems is performed under the same trust model. Second, weak bisimulation and congruence property are defined as in [van Glabbeek 2011]. Hence, since the semantic rules of parallel composition respect the format ensuring the congruence property [van Glabbeek 2011], the result immediately follows.

We point out that, by definition of $\approx$, our notion of behavioral equivalence relating trust adaptive systems is sensitive to the identities of the agents interacting among them. This is a deliberate choice as in the setting of this paper the main objective of equivalence checking is the analysis of the trust relationships among individuals and of the impact of the trust model (and of related attacks to the trust model) upon the behavior of such individuals.

As mentioned, among its various application domains, equivalence checking based on weak bisimulation can be used for the evaluation of the robustness of trust models against the attacks discussed in Section 2. One of the typical formal approaches followed in security verification relies on the definition of the most general adversary (mga) that, composed in parallel with the system, aims at violating the security condition of interest [Aldini and Di Pierro 2004; Focardi and Gorrieri 2004; Martinelli and Petrocchi 2007]. Then, the system is secure if it behaves always the same in the absence and in the presence of such an adversary.

In our framework, the attacks are against the trust model in order to falsify the reputation of a chosen agent, let us say agent J. For this purpose, the mga for such a kind of malicious behavior executes the action $obs(J, v)$, see rule $(op2)$ of Table II, in all possible ways. Formally, the behavioral pattern for the mga is as follows:

$$\mathrm{MGA} \stackrel{\mathrm{def}}{=} \sum_{G.\, J \in G} \sum_{v \in \mathsf{T}} ent(G).obs(J, v).\mathrm{MGA}$$

where $G.\, J \in G$ stands for G such that $J \in G$. Thanks to its nondeterministic nature, such a process term represents the various types of attacks conducted with the aim of giving false trust opinions. Hence, by adding the mga to the system we can evaluate the effect of such attacks on the system functionalities. This is done through equivalence checking as follows.

Definition 3.5. Let $\mathcal{T}$ be a trust model and $A = (\mathcal{S}, S, \mathcal{G}, \mathcal{E})$ a trust adaptive system. If A and $(\mathcal{S} \cup \{\mathrm{Adv} : \mathrm{MGA}\}, S, \mathcal{G}, \mathcal{E})$ are behaviorally equivalent under $\mathcal{T}$, then A under $\mathcal{T}$ is secure with respect to the mga Adv.

Notice that the two systems under comparison in Def. 3.5 differ only for the internal actions performed by the adversary to affect the trust model. Thus, if they do not satisfy the equivalence check, it means that the interference of the adversary impaired the trust model in such a way to modify the observable behavior of the trust adap-

tive system. For a comprehensive analysis of the trust model, however, it is also worth considering insider attacks conducted by a set $X \subseteq \mathcal{S}$ of agents involved in the community activities. For each agent $I : P$ in X, we define a new behavioral pattern P^{adv} obtained by changing P in a way inspired by the mga described above. First, each process term $obs(v).B$ is replaced by $\sum_{v \in \mathsf{T}} obs(v).B$. Second, each remaining process term $B \stackrel{\mathrm{def}}{=} P$ is replaced by $B \stackrel{\mathrm{def}}{=} P + \sum_{G.J \in G} \sum_{v \in \mathsf{T}} ent(G).obs(J, v).B$. Then, we replace the set $X = \{I_1 : P_1, \ldots, I_n : P_n\}$ with the set $X^{\mathrm{adv}} = \{I_1 : P_1^{\mathrm{adv}}, \ldots, I_n : P_n^{\mathrm{adv}}\}$ and apply the equivalence check to the two versions of the system.

Definition 3.6. Let $\mathcal{T}$ be a trust model and $A = (\mathcal{S}, S, \mathcal{G}, \mathcal{E})$ a trust adaptive system such that $X = \{I_1 : P_1, \ldots, I_n : P_n\} \subseteq \mathcal{S}$. If A and $(\mathcal{S}[X^{\mathrm{adv}}/X], S, \mathcal{G}, \mathcal{E})$ are behaviorally equivalent under $\mathcal{T}$, then A under $\mathcal{T}$ is secure with respect to X.

We emphasize that the attacker model we considered is general, as it takes into account any kind of trust falsification. It is possible to restrict ourselves to assume only certain liar strategies, like false positive opinions (if we are interested in the ballot stuffing attack) or false negative opinions (if, instead, the objective is to study the bad mouthing attack). For this purpose, it is sufficient to limit accordingly the portion T' of the domain T to consider when modeling the attack, and replace T with T' in the definition of MGA. The same approach can be applied to the modelling of insider attacks.

As an additional contribution, equivalence checking can be effectively used also to compare different trust models applied to a given trust adaptive system. The abstraction level of the behavioral equivalence is sufficient to take into account the impact of the trust model upon the observable system behavior without considering the details of the trust model, which are hidden at the semantics level. Therefore, two trust models could be perceived as indistinguishable if they do not induce different behaviors in the adaptive trust system to which they are applied.

Definition 3.7. Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two trust models, and $A = (\mathcal{S}, S, \mathcal{G}, \mathcal{E})$ a trust adaptive system. Let M_1 with initial state q_0 and M_2 with initial state q_0' be the TLTSs representing the semantics of A under $\mathcal{T}_1$ and A under $\mathcal{T}_2$, respectively. Then, $\mathcal{T}_1$ and $\mathcal{T}_2$ are equivalent from the viewpoint of A, denoted $A \vdash \mathcal{T}_1 \approx \mathcal{T}_2$, if $q_0 \approx q_0'$.

As emphasized, $\mathcal{T}_1$ and $\mathcal{T}_2$ may represent two different trust models, and in such a case the objective is to compare their strategies. On the other hand, $\mathcal{T}_1$ and $\mathcal{T}_2$ may represent two versions of the same trust model differing only for some configuration parameters, and in such a case the objective is to conduct sensitive analysis guided by such parameters.

In general, we conclude by observing that since two states labeled with different trust information may be bisimilar, weak bisimulation based minimization offers an interesting way of studying the actual impact of the policies and parameter values used by the trust model.

3.5. Logic of Trust and Model Checking

The equivalence checking based approach described above abstracts from the quantitative details of the trust model as the objective is to estimate its impact upon the observable behavior of the trust adaptive system. This is not sufficient whenever the result of the equivalence check is negative, as in such a case it is difficult to identify the behaviors causing the distinguishability result. Logics and model checking represent an alternative approach to the verification of system properties that is very popular in the formal methods literature, especially in the setting of security analysis [Armando et al. 2009; Halpern and Pucella 2012]. In this section, we introduce a logic for ex-

pressing properties of trust adaptive systems and trust models. To this aim, in [Aldini 2015], a model checking based approach is defined that relies on a trust temporal logic, called TTL, which is defined for the verification of TLTS-like models. Here, we adapt such a framework to the TLTS model.

The logic for TLTSs takes into account not only the system activities labeling the transitions but also the trust opinions associated with the states. For this reason, it is strictly more expressive than the nondeterministic equivalence used in the previous section. Similarly as for other logics merging action/state based predicates, atomic formulas include actions α labeling the TLTS transitions and state-based trust predicates of the form:

$$w \geq k$$

where $k \in \mathsf{T}$ and w is a trust variable, which is defined in two possible ways:

— $w = t_{IJ}$, i.e., the trust of I towards J as computed by the trust model $\mathcal{T}$;
— $w = f\{\!|v \,|\, p((J,v)_I) = true|\!\}$, where p is a boolean predicate applied to filter trust opinions (e.g., based on I, J, and v) and f is any associative and commutative function defined over the set of trust values. For instance, it could be sum, min, and count, provided that T is a domain of numbers, like $\mathsf{T} \subseteq \mathbb{Z}$ or $\mathsf{T} = [-1;1]$.

Therefore, an atomic state-based statement is a predicate about either the trust between agents as computed by the trust model $\mathcal{T}$, or the set of trust opinions reported by the agents.

The syntax of TTL is defined as follows:

$$\begin{array}{rcl} \Phi & ::= & true \mid \alpha \mid w \geq k \mid \Phi \wedge \Phi \mid \neg\Phi \mid A\pi \mid E\pi \\ \pi & ::= & \Phi \,{}_{\mathcal{A}_1}U\, \Phi \mid \Phi \,{}_{\mathcal{A}_1}U_{\mathcal{A}_2}\, \Phi \end{array}$$

where state formulas are ranged over by Φ, and include atomic statements, classical logical combinations of state formulas, and quantified path formulas; the universal (A) and existential (E) path quantifiers apply as usual to path formulas (π); path formulas include two flavors of the indexed until operator, where $\mathcal{A}_1, \mathcal{A}_2$ are subsets of the set L of labels of the TLTS.

Intuitively, a state satisfies an action-based predicate α if it enables a transition labeled with α. On the other hand, a state satisfies a state-based predicate $w \geq k$ if the evaluation of w in the state satisfies the condition $\geq k$. Notice that the evaluation of w in a state q depends on the trust opinions and agent groups labeling q. In the following, we use the notation w_q to express the value of variable w as computed in q. A path satisfies the until formula $\Phi \,{}_{\mathcal{A}_1}U\, \Phi'$ if the path visits a state satisfying Φ', and visits states satisfying Φ while performing only actions in $\mathcal{A}_1$ until that point. Similarly, the until formula $\Phi \,{}_{\mathcal{A}_1}U_{\mathcal{A}_2}\, \Phi'$ is satisfied by a path if the path visits a state satisfying Φ' after performing an action in $\mathcal{A}_2$, and visits states satisfying Φ while performing only actions in $\mathcal{A}_1$ until that point. We observe that a path satisfying $\Phi \,{}_{\mathcal{A}_1}U_{\mathcal{A}_2}\, \Phi'$ must include a transition to a state satisfying Φ', while this is not required for $\Phi \,{}_{\mathcal{A}_1}U\, \Phi'$ if the initial state of the path satisfies Φ'. As a shorthand, we also use the classical *eventually* operator $F\Phi$ to stand for the path formula $true \,{}_{L}U\, \Phi$.

Formally, to define the semantics of TTL, we need to introduce some notation. A path σ is a (possibly infinite) sequence of transitions of the form:

$$q_0 \xrightarrow{\alpha_0} q_1 \ldots q_{j-1} \xrightarrow{\alpha_{j-1}} q_j \ldots$$

where $q_{j-1} \xrightarrow{\alpha_{j-1}} q_j \in R$ for each $j > 0$. Every state q_j in the path is denoted by $\sigma(j)$. Moreover, let $q_j \xrightarrow{\mathcal{A}} q_{j+1}$ if and only if $\alpha_j \in \mathcal{A} \subseteq L$. We denote with $Path(q)$ the set

Table III. Semantics of TTL

$q \models true$	*holds always*
$q \models \alpha$	*iff* $\exists q' : q \xrightarrow{\alpha} q' \in R$
$q \models w \geq k$	*iff* $w_q \geq k$
$q \models \Phi \wedge \Phi'$	*iff* $s \models \Phi$ *and* $s \models \Phi'$
$q \models \neg\Phi$	*iff* $s \not\models \Phi$
$q \models A\pi$	*iff* $\forall \sigma \in Path(q) : \sigma \models \pi$
$q \models E\pi$	*iff* $\exists \sigma \in Path(q) : \sigma \models \pi$
$\sigma \models \Phi\, {}_{\mathcal{A}_1}U\, \Phi'$	*iff* $\exists k \geq 0 :$ $\sigma(k) \models \Phi' \wedge (for\ all\ 0 \leq i < k : \sigma(i) \models \Phi \wedge \sigma(i) \xrightarrow{\mathcal{A}_1} \sigma(i+1))$
$\sigma \models \Phi\, {}_{\mathcal{A}_1}U_{\mathcal{A}_2}\, \Phi'$	*iff* $\exists k > 0 :$ $\sigma(k) \models \Phi' \wedge (for\ all\ 0 \leq i < k-1 : \sigma(i) \models \Phi \wedge$ $\sigma(i) \xrightarrow{\mathcal{A}_1} \sigma(i+1)) \wedge \sigma(k-1) \models \Phi \wedge \sigma(k-1) \xrightarrow{\mathcal{A}_2} \sigma(k)$

of paths starting in state $q \in Q$. Then, the formal semantics of TTL is as reported in Table III.

Analogously as shown in [Aldini 2015], TTL can be mapped to the logic UCTL [ter Beek et al. 2008], for which an efficient on-the-fly model checking algorithm is defined.

TTL based model checking emphasizes that, even if the algebraic specification language is purely nondeterministic, quantitative analysis (in particular, sensitive analysis based on trust) is still possible. The measurements obtained in this way are useful to estimate the effectiveness and efficiency of the trust model (e.g., only honest and cooperative agents shall become trustworthy agents, as rapidly as possible), as well as the related robustness against the taxonomy of attacks surveyed in Section 1. In the next sections, examples of formula templates useful to guide such analyses will be provided depending on the specific trust models.

4. USE CASE: EVALUATING TRUST-INCENTIVE SERVICE MANAGEMENT

Trust-Incentive Service Management [Zhang et al. 2007] (TIM) is a framework mixing trust and pricing mechanisms to incentivize collaboration among agents in peer-to-peer (P2P) environments. TIM is characterized by a two-tier P2P architecture, where at the higher level we have special agents, called CDSRs, managing *clubs* of agents working at the lower level. The notion of club is used to aggregate multiple self-organizing peers with common needs/features in order to improve the efficiency of service discovery/delivery. Each club includes both service consumers and providers of the unique type of service associated with the club. Agents can (and are stimulated to) play both roles. Trust towards an agent is updated whenever the agent behaves as provider, by estimating the quality of the service delivered as well as the willingness of the agent to collaborate as service provider. Trust towards an agent is used to establish whether the agent can receive the service requested when behaving as consumer. Trust is generalized to express relations among clubs. The trust from club X to club Y, reporting the result of direct experiences among peers belonging to the two clubs, depends on the amount of positive experiences p and negative experiences n observed by peers in X when interacting with peers in Y:

$$t_{XY}(p,n) = \begin{cases} 1-\lambda^{p-n} & if\ p > n \\ 0 & otherwise \end{cases} \quad (1)$$

where $\lambda \in]0;1[$. The reputation of peer $K \in Y$ as perceived by the other peers of Y is:

$$t_{YK}(q,q') = \begin{cases} 1-\lambda^{q} & if\ q' = 0 \\ 0 & otherwise \end{cases} \quad (2)$$

where q (resp., q') is the number of positive (resp., negative) interactions between peers of Y and K. By combining these trust values, we obtain the trust of any peer in club X towards peer K belonging to another club Y:

$$t_{XK} = 1 - (1 - t_{YK})^{t_{XY}}. \tag{3}$$

All the interactions governed by trust are based on the following encoding of the trust model of [Zhang et al. 2007]. First, by virtue of the assumptions above, the trust opinion reported as feedback through action obs is either 1 or -1, respectively.

Then, on one hand, given clubs X and Y, the estimation of Eq. (1) is based on the following encoding of p:

$$p = \sum_{i \in X, j \in Y} mul((j,1)_i)$$

where $mul(e)$ denotes the multiplicity of term e in $\mathcal{E}$. We argue similarly in the case of parameter n, by replacing 1 with -1 in the equation above. On the other hand, in the case of Eq. (2), given agent k in the club Y, parameter q is determined as follows:

$$q = \sum_{j \in Y, j \neq k} mul((k,1)_j)$$

and, analogously, the case of negative interactions is evaluated by testing the condition $\sum_{j \in Y, j \neq k} mul((k,-1)_j) = 0$.

As far as the behavior of the agents is concerned, the system includes behavioral patterns for the three categories surveyed above: consumers, providers, and CDSRs. For the sake of simplicity, we just consider two service types and, therefore, two clubs.

The behavioral patterns are defined as follows:

$$\begin{array}{rcll}
Cons & \stackrel{\text{def}}{=} & req_srv_1.rcv_cmt_1.Cons_1 + req_srv_2.rcv_cmt_2.Cons_2 & \\
Cons_i & \stackrel{\text{def}}{=} & rcv_srv_i.obs(1).Cons + & \\
 & & rcv_srv_i_dnl.obs(-1).Cons & i \in \{1,2\} \\
CDSR_i & \stackrel{\text{def}}{=} & rcv_req_srv_i.fwd_req_i.snd_cmt_i.CDSR_i + & \\
 & & rcv_req_srv_{3-i}.fwd_req_CDSR_{3-i}.rcv_ack_req_{3-i}.snd_cmt_{3-i}.CDSR_i + & \\
 & & rcv_fwd_CDSR_i.fwd_req_i.snd_ack_req_i.CDSR_i & i \in \{1,2\} \\
Prov_i & \stackrel{\text{def}}{=} & rcv_req_i.(snd_srv_i.Prov_i + snd_srv_i_dnl.Prov_i) & i \in \{1,2\}
\end{array}$$

The synchronization set contains the pairs $req\_srv_i \times rcv\_req\_srv_i$, $fwd\_req_i \times rcv\_req_i$, $snd\_cmt_i \times rcv\_cmt_i$, $fwd\_req\_CDSR_i \times rcv\_fwd\_CDSR_i$, $snd\_ack\_req_i \times rcv\_ack\_req_i$, $snd\_srv_i \times rcv\_srv_i$, $snd\_srv_i\_dnl \times rcv\_srv_i\_dnl$, with $i \in \{1,2\}$. Intuitively, the consumer sends the request of the chosen service type i (req_srv_i), waits for a commit (rcv_cmt_i), and then receives either the service (rcv_srv_i) or the refusal ($rcv\_srv_i\_dnl$). The result of the interaction is then estimated in terms of trust feedback. The CDSR receives service requests, by forwarding those related to the local service directly to a provider (fwd_req_i) and those related to the remote service to the other CDSR ($fwd\_req\_CDSR_i$), which acknowledges the request ($snd\_ack\_req_i$). The provider receives the requests and decides, based on trust ($snd\_srv_i \in H$ and $snd\_srv_i\_dnl \in L$), whether to collaborate. For the sake of brevity, we do not report the behavior of agents switching from consumer to provider and vice versa.

Some comments about usability and scalability are in order. First, it is worth noticing that the behavioral patterns shown above are not affected by the system topology. All information concerning membership of clubs and interactions among specific agents (e.g., between a consumer and the CDSR of the related club) is left to the definition of synchronization set and communicating groups. Similarly, the assignment of

the trust opinions to specific agents is governed by the semantics rules in a transparent way. More precisely, the trust feedback, reported through action obs, is assigned to the unique provider interacting with the consumer by virtue of the semantics rules of Table II. As a consequence, the specification of the behavioral patterns is compact, intuitive and does not depend on the size of the system and on the complexity of the underlying interactions.

The topology under analysis includes two consumers, $C_1 : Cons$ and $C_2 : Cons$, two providers $P_1 : Prod_1$ and $P_2 : Prod_2$, and two clubs, governed by $D_i : CDSR_i$, with $i \in \{1, 2\}$, respectively. The first, resp. second, club is represented by group $Club_1 = \{C_1, P_1, D_1\}$ (resp. $Club_2 = \{C_2, P_2, D_2\}$). Moreover, in order to enable the direct interactions among consumers and providers, we also have the groups $\{C_i, P_j\}$, with $i, j \in \{1, 2\}$. In the following, we use L to denote the set of all possible actions labeling the transitions of the TLTS underlying such a system.

4.1. Formal verification

We start the verification of TIM through model checking.

In a first, ideal scenario, it is worth verifying the way in which the trust model works whenever all the agents behave honestly and are cooperative. This is done by studying the relation existing between the behavior of a honest agent and the trust towards such an agent estimated by the other agents of the system. In particular, we analyze how the trust towards a producer as perceived outside the related club is determined depending on the services delivered by such an agent inside and outside its club. The specific parameters we consider are the number of services delivered by P_2, call it ns, and the trust $t_{Club_1 P_2}$ towards P_2 as perceived by $Club_1$. Since all the consumers are honest, it turns out that parameter ns is given by $\sum_{i \in \{C_1, C_2\}} mul((P_2, 1)_i)$. Formally, the relation above is investigated by model checking the following formula:

$$EF(\sum_{i \in \{C_1, C_2\}} mul((P_2, 1)_i) = k \wedge t_{Club_1 P_2} > w)$$

which is true in a state q if a path departing from q eventually visits a state in which the two atomic state-based predicates are satisfied. According to the formal definition of Section 3.5, the former is a predicate about filtered trust opinions checked against the threshold value k, while the latter is a predicate about a trust metric checked against the threshold value w.

To emphasize the nature of the relation among the parameters of interest, we report in Fig. 1 the results of a study conducted on the metric $t_{Club_1 P_2}$. By varying ns (see the horizontal axis) and parameter λ of the TIM model, we show the maximum value of such a metric – curves (a) and (b) – and the minimum non-zero [5] value of it – curves (c) and (d). Under the same configuration of the parameters, the variability of the trust metric between a minimum and a maximum depends on the nondeterminism in the choice of the service requests, which are either issued locally to the same club of the requester or issued remotely to an external club. Hence, for each experiment, we also specify the distribution of local and remote requests. For instance, given $\lambda = 0.75$ and $ns = 6$, the maximum value of $t_{Club_1 P_2}$ is 0.38, which is obtained whenever 4 local requests and 2 remote requests are served by P_2. The minimum trust is always obtained whenever only 1 request is local. Hence, these results show how $t_{Club_1 P_2}$ is influenced differently from local and remote requests to P_2. The shape of the curves reveals also that by increasing the number of observations the trust values increase as well to-

[5] By definition, $t_{Club_1 P_2} > 0$ only if P_2 serves at least one local and one remote request, which represents the condition we assume to compute the minimum non-zero value of $t_{Club_1 P_2}$.

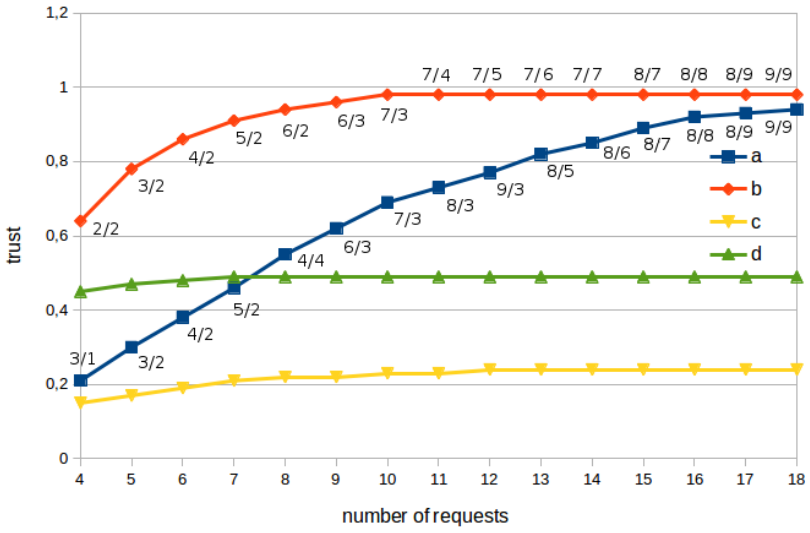

Fig. 1. TIM trust analysis - case (a): maximum trust and $\lambda = 0.75$, case (b): maximum trust and $\lambda = 0.5$, case (c): minimum trust and $\lambda = 0.75$, case (d): minimum trust and $\lambda = 0.5$.

wards limiting values that depend on the specific model parameters. In particular, the curves related to minimum trust converge towards the value $1 - \lambda$, while curve (a) reaches the same limiting value of curve (b) only when $ns \geq 25$, thus revealing the influence of parameter λ on the promptness (and, symmetrically, the cautiousness) of the trust model to react to the observed behaviors.

Since the source of the requests has an important effect on the trust computation, we then show the effects of agents' migration. More precisely, we change the behavior of the consumer, by adding to the behavioral pattern $Cons$ the expression:

$$+ent(Club_i) + esc(Club_i) \quad i \in \{1, 2\}$$

stating that consumers can dynamically change their reference club. In order to favor a comparison based on equivalence checking between such a dynamic scenario and the previous, static version of the system, we add to the provider behavioral pattern also the consumer behavior. The objective is to check whether the trust gained by each provider is then enough to get access to the service of the other provider. The result is that for every positive value of the trust threshold th the two versions of the system are not behaviorally equivalent under the TIM trust model. The version with static clubs does not allow P_1 to obtain services from P_2 if P_1 does not serve local requests of C_1, as in such a case the direct trust from $Club_1$ to P_1 is null, and so remains also the trust from $Club_2$ to P_1.

Formally, the property:

$$E(true\ {}_{\mathcal{A}_1}U\ t_{Club_2P_1} > 0)$$

where $\mathcal{A}_1 = L\backslash\{P_1.snd\_srv_1 \times C_1.rcv\_srv_1\}$, holds if a state satisfying $t_{Club_2P_1} > 0$ is reachable through a path along which the action $P_1.snd\_srv_1 \times C_1.rcv\_srv_1$ is never performed. Such a property is satisfied by the dynamic version but not by the static version of the system. Intuitively, this reveals a major degree of flexibility of the TIM model in case of agents' migration.

After having investigated the general behavior of the trust model, we consider its robustness against agents' attacks. By following the noninterference approach discussed

in Section 3.4, we evaluate the impact of consumers behaving as insider attackers with the specific aim of conducting a bad mouthing attack (against P_1). Again, for every positive value of the trust threshold th, the result we obtain is that the TIM model does not meet the equivalence check, as in the version under attack eventually P_1 may become untrusted. Formally, this condition is stated by the following property:

$$EF(\neg EF(t_{Club_2 P_1} > 0))$$

which intuitively means that it is reachable a state starting from which for every possible future, $t_{Club_2 P_1}$ is invariantly equal to zero. The reason is that it is quite simple to compromise the reputation of a provider by injecting a negative local opinion by a consumer belonging to the same club of the provider. Such a kind of malicious behavior suggests to enrich the CDSR functionalities to monitor the correctness of the feedback reported by every local agent. Hence, it becomes interesting to evaluate the effects of negative remote opinions reported by agents belonging to other clubs. For this purpose, similarly as in the case of Fig. 1, we perform a model checking analysis based on the following formula:

$$EF(\sum_{i\in\{C_1,C_2\}} mul((P_2,1)_i) = k \wedge mul((P_2,-1)_{C_1}) = n \wedge t_{Club_1 P_2} > w)$$

Parameter k denotes the number of positive opinions reported about P_2 and parameter n represents the number of negative opinions about P_2 reported by C_1. The results of the sensitive analysis are shown in Fig. 2 in the case of the maximum trust. They reveal the relation among parameter k (represented in the horizontal axis), the number n of negative opinions (ranging from 1 to 3, see the three different curves for each value of λ), and the maximum value of the trust metric $t_{Club_1 P_2}$ (see the vertical axis). As expected, notice that the convergence speed observed in Fig. 1 slows down because of the negative opinions.

We recall that the noninterference analysis conducted above emphasizes the impact upon provider's reputation of negative opinions resulting from the behavior of malicious consumers. By complementing the class of insider attackers, i.e., by assuming honest consumers and selfish providers that do not trust the consumers, even if the interpretation changes we still obtain the same results commented above. In other words, the relation among the considered trust parameters is always the same independently of the motivations – correct or not – behind the negative opinions.

In order to integrate the analysis above, we consider the ballot stuffing attack conducted by the consumers. Again, the equivalence check is not satisfied. For instance, by setting $\lambda = 0.5$ and $th = 0.3$, it holds that two services delivered by P_2 are not sufficient for P_2 to get access to any service in the ideal scenario, while they are if a ballot stuffing attack is conducted locally in $Club_2$ to falsify the reputation of P_2. To illustrate more clearly the effects of the attack, we reconsider the analysis of Fig. 1 in the case $\lambda = 0.75$. Then, we compare the original results, see curves (a) and (c) in Fig. 3, with those obtained in a scenario in which C_2 performs the ballot stuffing attack to favor P_2, see curves (b) and (d) of Fig. 3.

In general, it is worth emphasizing that the TIM model has been designed with the aim of combining orthogonal incentive mechanisms deriving from reputation and from remuneration. Several monitoring activities are left underspecified and to the design of the CDSR layer, in which potential misbehaviors can be detected. Hence, the TIM model does not concentrate on the vulnerabilities of the trust infrastructure. However, a careful analysis of such vulnerabilities, as emphasized by the verification results presented in this section, reveals that compromising the trust model is possible, even easily, in a way that may jeopardize the whole incentive infrastructure.

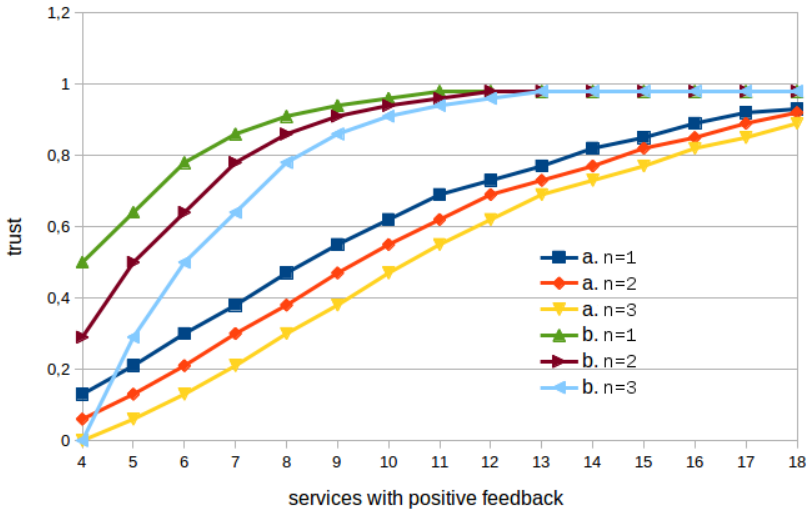

Fig. 2. TIM maximum trust analysis with bad mouthing - case (a): $\lambda = 0.75$, case (b): $\lambda = 0.5$.

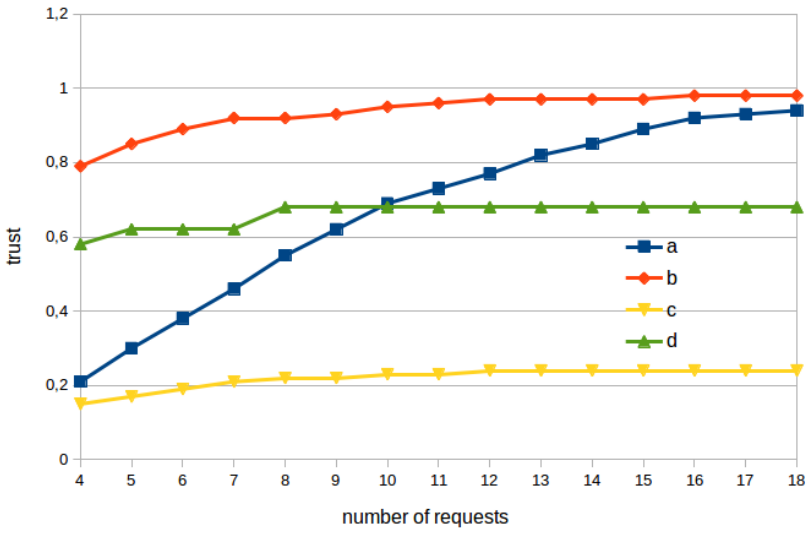

Fig. 3. TIM trust analysis ($\lambda = 0.75$) - case (a): maximum trust without attack, case (b): maximum trust with ballot stuffing, case (c): minimum trust without attack, case (d): minimum trust with ballot stuffing.

5. USE CASE: COMPARISON BETWEEN RFSN AND RRS

Several frameworks have been proposed for developing a web of trust in sensor networks and mobile ad-hoc networks in a fully distributed way. We consider two of them in which agents maintain reputation information about their neighbors by monitoring their cooperative and non-cooperative behaviors. Both direct observations and second-hand recommendations represent parameters used to update reputation. Then, trust is estimated as the statistical expectation of a probability distribution representing such a reputation. More precisely, based on a Bayesian approach, trust is determined

by an uncertain probability drawn according to a distribution that depends on reputation. The most used reputation function is the Beta distribution $\mathrm{Beta}(\alpha, \beta)$, and trust turns out to be the statistical expectation of such a function:

$$\mathtt{E}(\mathrm{Beta}(\alpha, \beta)) = \frac{\alpha}{\alpha + \beta}.$$

The result of the comparison between such a trust metric and the trust threshold th is then used to govern the decision to cooperate.

The two systems under analysis are the Reputation-based Framework for Sensor Networks [Ganeriwal et al. 2008] (RFSN) and the Robust Reputation System [Buchegger and Boudec 2004] (RRS) integrated in the CONFIDANT protocol [Buchegger and Le Boudec 2002]. Basically, they differ for the way in which the two reputation parameters α and β are computed/updated and for the reputation sharing policy. Here, we model and compare these systems with respect to the typical scenario of the Dynamic Source Protocol [Johnson et al. 2001] (DSR), in which agents require (and then monitor) message forwarding operations to their neighbors.

In RFSN, α and β represent the numbers of cooperative and non-cooperative behaviors, respectively, observed directly through a watchdog mechanism. The reputation function is $\mathrm{Beta}(\alpha + 1, \beta + 1)$ and, initially, $\alpha = \beta = 0$, so that $\mathrm{Beta}(1, 1)$, corresponding to a uniform distribution on the measureable space, represents the initial uncertainty due to the absence of any prior information. RFSN enables the application of an aging mechanism, according to which recent observations are given more weight with respect to old observations. For instance, whenever updating α by adding r observations, the new value is computed as $\alpha = (w \cdot \alpha) + r$, where $w \in [0, 1]$ is the aging weight. The reputation of J from the viewpoint of I can take into account the recommendation about J provided by a third agent K. Denoted α_{IJ} the parameter expressing the number of cooperative behaviors of J observed by I in direct interactions between them, the value combining direct experience with second-hand information is:

$$\alpha_{IJ} = \alpha_{IJ} + \frac{2 \cdot \alpha_{IK} \cdot \alpha_{KJ}}{((2 + \beta_{IK}) \cdot (2 + \alpha_{KJ} + \beta_{KJ})) + 2 \cdot \alpha_{IK}}$$

and, analogously:

$$\beta_{IJ} = \beta_{IJ} + \frac{2 \cdot \alpha_{IK} \cdot \beta_{KJ}}{((2 + \beta_{IK}) \cdot (2 + \alpha_{KJ} + \beta_{KJ})) + 2 \cdot \alpha_{IK}}.$$

To avoid the bad-mouthing vulnerability, negative recommendations resulting in trust values below the chosen trust threshold are not accepted. Moreover, to avoid propagation of redundant information, only the values α and β deriving from direct observations can be recommended to other parties. Notice that no recommendations of the form $\alpha = \beta = 0$ are shared, as they do not represent any direct observation.

In RRS, α and β represent the numbers of non-cooperative and cooperative behaviors, respectively. Since they have the opposite interpretation with respect to RFSN, every relation based on the trust metric is to be inverted. For instance, any decision about cooperation is based on the test $\mathtt{E}(\mathrm{Beta}(\alpha, \beta)) < th$.

The default initial value of the reputation parameters is $\alpha = \beta = 1$. The same aging mechanism surveyed above can be applied. Reputation of J as perceived by I is updated when direct observations of I about J occur or when indirect observations from K about J are received and accepted by I. In the latter case, the update $\alpha_{IJ} = \alpha_{IJ} + w' \cdot \alpha_{KJ}$ is applied, with w' a small positive constant weighting the recommendation. The same holds for parameter β. Updates are accepted if one of the two following conditions holds:

— K is *trustworthy*;
— K is *untrustworthy*, but the deviation test $|\mathtt{E}(\mathrm{Beta}(\alpha_{IJ},\beta_{IJ}))-\mathtt{E}(\mathrm{Beta}(\alpha_{KJ},\beta_{KJ}))| < d$ holds, with d a small positive constant denoting the precision of the test.

The trustworthiness condition about K is satisfied if $\mathtt{E}(\mathrm{Beta}(\gamma,\delta)) < th$ holds, where γ and δ count the number of times in which the deviation test about K failed and succeeded, respectively (initially, $\gamma = \delta = 1$). In other words, trustworthiness is estimated by the same Bayesian approach used for trust. However, untrustworthy agents are not punished in terms of reputation.

In our modeling framework, the number of cooperative and non-cooperative behaviors of J observed by I is given by $mul((J,1)_I)$ and $mul((J,-1)_I)$, respectively, because every action of the form $obs(1)$ (resp. $obs(-1)$) represents the observation of a cooperative (resp. non-cooperative) behavior.

As far as the behavior of the agents is concerned, the system includes the behavioral patterns modeling the several roles of the DSR forward service:

$$\begin{aligned}
\mathit{Requestor} &\stackrel{\text{def}}{=} \mathit{snd_req}.(\mathit{rcv_acc}.\mathit{obs}(1).\mathit{Requestor} + \mathit{rcv_dnl}.\mathit{obs}(-1).\mathit{Requestor})\\
\mathit{Forwarder} &\stackrel{\text{def}}{=} \mathit{rcv_req}.(\mathit{snd_acc}.\mathit{Forward} + \mathit{snd_dnl}.\mathit{Forwarder})\\
\mathit{Forward} &\stackrel{\text{def}}{=} \mathit{snd_fwd}.\\
&\quad (\mathit{rcv_acc}.\mathit{obs}(1).\mathit{Forwarder} + \mathit{rcv_dnl}.\mathit{obs}(-1).\mathit{Forwarder})\\
\mathit{Recipient} &\stackrel{\text{def}}{=} \mathit{rcv_fwd}.(\mathit{snd_acc}.\tau.\mathit{Recipient} + \mathit{snd_dnl}.\tau.\mathit{Recipient})
\end{aligned}$$

Synchronizations are as follows: $\mathit{snd\_req} \times \mathit{rcv\_req}$, $\mathit{snd\_acc} \times \mathit{rcv\_acc}$, $\mathit{snd\_dnl} \times \mathit{rcv\_dnl}$, $\mathit{snd\_fwd} \times \mathit{rcv\_fwd}$, $\mathit{snd\_fwd} \times \mathit{rcv\_req}$.

A requestor agent generates the message to be sent to the final recipient through a path of forwarding agents. The choice of the path is trust-based ($\mathit{snd\_req}, \mathit{rcv\_req}, \mathit{snd\_fwd}, \mathit{rcv\_fwd} \in H$) as well as the decision about whether to forward a request ($\mathit{snd\_acc}, \mathit{rcv\_acc} \in H$, while $\mathit{snd\_dnl}, \mathit{rcv\_dnl} \in L$).

Topologically, we consider a system A composed of four agents $\mathit{Req}_1, \mathit{Req}_2 : \mathit{Requestor}$, $\mathit{For} : \mathit{Forwarder}$ and $\mathit{Rec} : \mathit{Recipient}$, such that the communicating groups are $\{\mathit{Req}_1, \mathit{Req}_2, \mathit{For}\}$ and $\{\mathit{For}, \mathit{Rec}\}$.

5.1. Formal verification

The preliminary analysis we perform emphasizes the indistinguishability between the two trust models in the absence of non-cooperative behaviors. Indeed, if RFSN and RRS employ equivalent trust thresholds (formally, $th_{\mathrm{RFSN}} = 1 - th_{\mathrm{RRS}}$) and if all the agents are not selfish and/or malicious, then it turns out that they are equivalent from the viewpoint of the system described above. Formally, $A \vdash \mathrm{RFSN} \approx \mathrm{RRS}$.

Even if they are equivalent, it is worth comparing their efficiency through model checking in terms of convergence speed towards the best trust value, which is 1 in the case of RFSN and 0 in the case of RRS. The results, shown in Fig. 4, refer to the best trust value from agent Req_1 to agent For resulting after a given number of requests, see curves (a) for RFSN and (b) for RRS. In general, they reveal that the convergence for RFSN is slightly more rapid. The difference is due to the recommendation policies used to manage the second-hand information received from Req_2. In the case of RRS, we recall that the trustworthiness of Req_2 (and of its recommendations) depends on the satisfiability of the deviation test, which is a result hard to achieve whenever the frequencies with which Req_1 and Req_2 interact with For are different. Hence, curve (b), which refers to the best case scenario, may be too optimistic with respect to realistic situations. In particular, it is associated with a scenario in which Req_2 recommends highly positive values whenever it is trusted by Req_1 and recommends values compatible with the reputation of For computed by Req_1 whenever it is not trustworthy. To

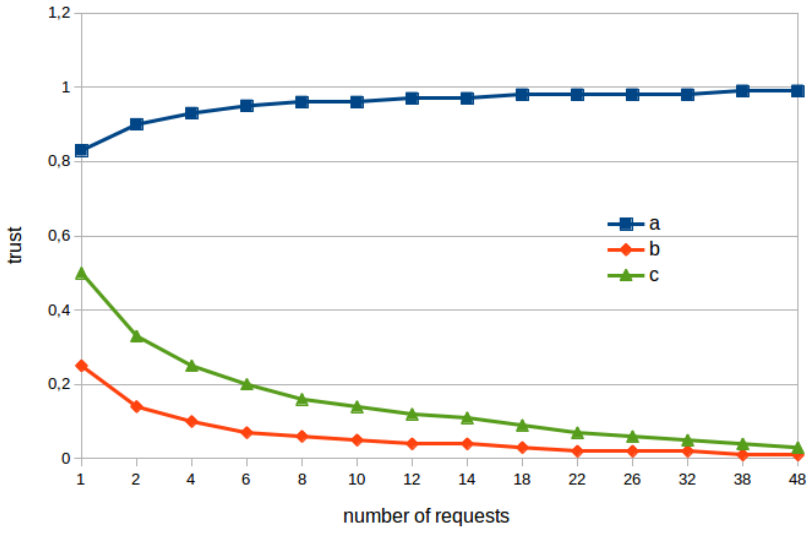

Fig. 4. RFSN (a) and RRS (b-c) trust analysis: comparison.

emphasize such an aspect, we report also curve (c) showing the performance whenever the deviation test is never satisfied by Req_2.

To extend the comparative analysis, it is worth considering the two trust models whenever at least one agent is either malicious or selfish.

Assume that agent Req_2 executes a bad mouthing attack against agent For in order to convince agent Req_1 to discard For from its path. For this purpose, the algebraic specification of agent Req_2 is changed as discussed in Section 3.4. The related equivalence check is positive in the case of RFSN, i.e., the trust model is robust against false recommendations provided by Req_2. Instead, the result is negative for RRS. As investigated through model checking, the property stating that there exists a state starting from which the action $Req_1.snd\_req$ is never enabled:

$$EF(\neg EF(Req_1.snd\_req)) \tag{4}$$

is satisfied by the version with malicious Req_2, but not by the original version. The reason revealed by the counterexample is quite subtle. Assume $\frac{1}{2} + \epsilon < th < \frac{2}{3}$ so that $\mathtt{E}(\mathrm{Beta}(1,1)) = 0.5 < th$, i.e., initially, the trust based check is satisfied by every agent. Hence, the first recommendation sent by Req_2 to Req_1 is accepted independently of the result of the deviation test. If such a recommendation is received before the first direct interaction between Req_1 and For and it is very negative, with $\alpha_{Req_2 For}$ sufficiently high, then, even for a small weighting factor w', it is possible to obtain $\alpha_{Req_1 For} = 2$, which causes $t_{Req_1 For} = \frac{2}{3}$ and, therefore, $t_{Req_1 For} > th$.

Now, assume that the misbehaving agent is For, which may consider the requestors as untrusted agents, thus denying the forwarding service. The same effect would be caused in the case For is simply a selfish agent. Obviously, through the equivalence check of Section 3.4, we derive that the original system is not equivalent to such a version. For both trust models, $t_{Req_1 For}$ converges towards the worst trust value by virtue of the non-cooperative behavior of agent For. Hence, there exists a state starting from which Req_1 does not send requests to For anymore. Formally, Property 4 is satisfied. This is a typical case in which the effectiveness of the trust model is demonstrated by a negative equivalence check. In order to detail such a result, we estimate through

model checking the efficiency of the trust models against the non-cooperative behavior. For RFSN, even with a low trust threshold (e.g., $th_{\mathrm{RFSN}} = 0.4$), in the worst case agent Req_1 sends just one request to agent For and, after the denial of service, eliminates For from its path. By applying the equivalent trust threshold in RRS, we obtain the same result. However, while in RFSN no recommendation can affect the decisions of agent Req_1, in RRS a recommendation from agent Req_2 may convince Req_1 to eliminate For even without any direct interaction. This behavior strictly depends on the value of the recommendation and of the weighting factor w', which may be used as tuning parameters to conduct sensitive analysis.

In order to refine the attack, assume a collusion between agents For and Req_2, and compare such a version with the previous one in which For was the unique misbehaving agent. Again, the equivalence check is negative. The new attack revealed through model checking is as follows. As in the previous case, agent For refuses all the requests from agent Req_1. However, now agent Req_2 executes a ballot stuffing attack in collusion with agent For by sending positive recommendations to agent Req_1.

In the case of RFSN, and with $th_{\mathrm{RFSN}} = 0.4$, if agent Req_2 is trusted by agent Req_1, then we obtain that Req_1 considers For to be a trusted partner in spite of its non-cooperative behavior. The condition $t_{Req_1 For} < th$ holds only after 14 requests sent by agent Req_1 to For and, even worse, Property 4 is not satisfied. In [Ganeriwal et al. 2008], it is claimed that in similar cases the agent receiving the unfair recommendation can neglect it and reduce the trust towards the recommender. However, the conditions leading to such a decision are not clearly stated and, in any case, require that agent Req_1 is able to evaluate the quality of the recommendations received, by monitoring (and estimating) the direct interactions between Req_2 and For, which is not always the case. Moreover, notice that the same counterexample described above applies whenever agents For and Req_2 are not colluding and For executes the on-off attack, by choosing to be cooperative with Req_2 but not with Req_1. In such a case, the recommendations provided by agent Req_2 are correct and, therefore, there is no reason to punish Req_2. Going back to the analysis of the on-off attack conducted by For, we also specify that there exists a strategy allowing For to be trusted by Req_1 infinitely often, even by refusing some of its requests infinitely often. Such an attack can be avoided by setting $th_{\mathrm{RFSN}} > 0.5$.

Thanks to the use of the deviation test, the RRS model suffers only partly the vulnerabilities discussed above. Under analogous conditions, and with $th_{\mathrm{RRS}} = 0.6$, agent Req_1 distrusts (definitely) agent For after 8 requests.

In general, the second-hand information management of RRS makes the model robust against falsified recommendations. However, the formal verification reveals that it is not easy to tune the configuration parameters in order to trade effectiveness of the model against liar strategies with the efficiency of the recommendation system. On the other hand, RFSN is efficient in the case of cooperative communities and in the case of bad mouthing attacks, but it suffer some vulnerabilities for more sophisticated attacks. Finally, we observe that the performance results have been obtained without applying the aging factor, which, similarly as for the other parameters, may be used to conduct sensitive analysis.

6. CONCLUSION AND RELATED WORK

Trust and reputation are at the base of lightweight, indirect incentive mechanisms stimulating cooperation in collaborative environments while preventing selfish and malicious behaviors. Their formulation and deployment requires a careful analysis of the effectiveness of the underlying information sharing and decision-making policies. The framework proposed to address these issues encompasses several different ingredients useful to take into account the adaptive and dynamic nature of the systems

under consideration and the variety and complexity of the security threats to which distributed trust models are exposed.

From the modeling standpoint, separation of concerns and abstraction of the mechanisms implementing the trust infrastructure are the key features behind the usability and flexibility of our approach. The trust model is only intended to describe how opinions are combined to create reputation and trust relationships, as the main goal is to estimate system effectiveness and robustness against the injection of fake information. All the technical issues concerning the distribution of such values and the security of their storage are out of the scope of our analysis framework.

The case studies emphasize the advantages of our formal approach, i.e., a systematic way for analyzing the effects of attacks to the trust model and a unifying framework for the comparison of different trust models. These results derive from the combined use of equivalence and model checking, thanks to which sensitive and tradeoff analysis can be conducted, along the same line of approaches like, e.g., [Aldini and Bernardo 2007]. To extend the verification capabilities, quantitative analysis based on probabilistic behaviors can be integrated in our framework, e.g., by following the same approach proposed in [Aldini 2015], where trust values are used as weights governing probabilistic choices.

In the literature, trust and reputation models are analyzed typically through ad-hoc simulation [Kim 2009; Ganeriwal et al. 2008; Zhang et al. 2007; Kamvar et al. 2003], sometimes by means of game theory [Li and Shen 2012], and, only in few cases, via model checking [Aldini 2015; Kwiatkowska et al. 2013; He et al. 2010; Reith et al. 2007]. In most cases, the focus is on the validation of new models, while less attention is devoted to the formal, exhaustive comparison among different models with respect to efficiency requirements and robustness against the attack taxonomy. One of the first proposals posing such a work is described in [Marmol and Perez 2011], where a Java-based simulator is used to compare the accuracy of four different models in terms of the percentage of success when selecting an actually trustworthy service provider. The analysis is conducted for a system of wireless sensors even with respect to collusion attacks based on bad mouthing and ballot stuffing. Simulation frameworks are also used in [Fung et al. 2009], which proposes the analysis of efficiency, robustness and scalability of trust models, and in [Schlosser et al. 2004], supporting the computation of global notions of reputation.

In [Jonker and Treur 1999], a mathematical framework defining trust evolution and update functions is proposed that supports formal analysis of the dynamics of trust. It does not consider concurrency and attack models. In [Chandrasekaran and Esfandiari 2015] a generic graph-based testbed for evaluating social trust models is proposed, with some limitations about expressiveness of the modeling capabilities. Other methodologies, based on formalizations of trust semantic structures and their logical properties, concentrate on the trust model rather than its deployment in formally specified, concurrent systems [Huang and Fox 2006; Nicol and Huang 2010; Trcek 2009; Muller 2010]. The theory of semirings [Theodorakopoulos and Baras 2006], equivalence checking [Martinelli 2005; Carbone et al. 2004], and game theory [Wang et al. 2016] are also proposed as formal frameworks for the analysis of trust models.

Explicit notions of dynamic environment governing the communication are present in several different formal languages, ranging from the Ambient Calculus [Cardelli and Gordon 2000] to the most recent proposals [Bortolussi et al. 2015; Luisa Vissat et al. 2016]. They represent good candidates for integrating the approach proposed in this paper. For instance, the explicit representation of the environment and the interaction model supporting local and global communication make CARMA [Bortolussi et al. 2015; Loreti and Hillston 2016] an ideal framework for encompassing the trust-based modeling infrastructure presented in this paper. In particular, CARMA agents

include both a behavioral part in terms of a process term and a store modeling the local knowledge of the agent. Since the process behavior can be affected by the store by means of the application of predicates guarding process terms, trust- and group-based communications encoding the semantics rules of Table II are possible. Moreover, thanks to the environment representation of CARMA based on sophisticated evolution rules, extended quantitative behaviors, like trust-based probabilistic choice, can be encoded in a natural way.

The model checking results of the experiments proposed in Sections 4 and 5 have been obtained by using NuSMV. The specification of the case studies behavioral patterns have been encoded in corresponding modules of the NuSVM specification language, which include explicitly also ad-hoc variables storing opinions and the behavior of the trust machinery needed to convert opinions and recommendations into trust values. The obtained finite state machines have a size ranging from 2^{17} to 2^{26} states. The automatic mapping to NuSMV is under development, while the encoding of our approach in alternative frameworks, like CARMA, is left as future work.

REFERENCES

A. Aldini. 2015. Modeling and verification of trust and reputation systems. *Journal of Security and Communication Networks* 8, 16 (2015), 2933–2946.

A. Aldini. 2016. A formal framework for modeling trust and reputation in collective adaptive systems. In *Workshop on Formal Methods for the Quantitative Evaluation of Collective Adaptive Systems (FORECAST)*, Vol. 217. EPTCS, 19–30.

A. Aldini, B. Bernardo, and F. Corradini. 2010. *A process algebraic approach to software architecture design*. Springer.

A. Aldini and M. Bernardo. 2007. A formal approach to the integrated analysis of security and QoS. *Reliability Engineering & System Safety* 92, 11 (2007), 1503–1520.

A. Aldini and A. Di Pierro. 2004. A quantitative approach to noninterference for probabilistic systems. In *Procs. of Formal Methods for Security and Time*, Vol. 99. ENTCS, 155–182.

A. Armando, R. Carbone, and L. Compagna. 2009. LTL model checking for security protocols. *Journal of Applied Non-Classical Logics* 19 (2009), 403–429.

M. Bernardo, R. De Nicola, and J. Hillston (Eds.). 2016. *Formal methods for the quantitative evaluation of collective adaptive systems*. LNCS, Vol. 9700. Springer.

T. Beth, M. Borcherding, and B. Klein. 1994. Valuation of trust in open networks. In *Conference on Computer Security*. Springer, 3–18.

L. Bortolussi, R. De Nicola, V. Galpin, S. Gilmore, J. Hillston, D. Latella, M. Loreti, and M. Massink. 2015. CARMA: Collective adaptive resource-sharing Markovian agents. In *13th Workshop on Quantitative Aspects of Programming Languages and Systems (QAPL)*, Vol. 194. EPTCS, 16–31.

S. Buchegger and J.-Y. Le Boudec. 2004. A robust reputation system for peer-to-peer and mobile ad-hoc networks. In *2nd Workshop on the Economics of Peer-to-Peer Systems (P2PEcon)*.

S. Buchegger and J.-Y. Le Boudec. 2002. Performance analysis of the CONFIDANT protocol. In *3rd ACM Int. Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc'02)*. ACM, 226–236.

M. Carbone, M. Nielsen, and V. Sassone. 2004. A calculus of trust management. In *Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'04) (LNCS)*, Vol. 3328. Springer, 161–173.

L. Cardelli and A. D. Gordon. 2000. Mobile ambients. *Theoretical Computer Science* 240, 1 (2000), 177–213.

P. Chandrasekaran and B. Esfandiari. 2015. Toward a testbed for evaluating computational trust models: experiments and analysis. *Journal of Trust Management* 2, 8 (2015).

J.-H. Cho, A. Swami, and I.-R. Chen. 2011. A survey on trust management for mobile ad hoc networks. *Communications Surveys & Tutorials* 13, 4 (2011), 562–583.

A. Cimatti and others. 2002. NuSMV 2: An opensource tool for symbolic model checking. In *14th Conf. on Computer Aided Verification (CAV)*, Vol. 2404. LNCS, 359–364.

R. Focardi and R. Gorrieri. 2004. Classification of security properties - Part II: Network security. In *FOSAD 2001/2002 Tutorial Lectures*. LNCS, Vol. 2946. Springer, 139–185.

W. Fokkink. 2007. *Introduction to process algebra*. Springer.

C. Fung, J. Zhang, I. Aib, R. Boutaba, and R. Cohen. 2009. Design of a simulation framework to evaluate trust models for collaborative intrusion detection. In *Proc. of Network and Service Security (N2S'09)*. IEEE.

S. Ganeriwal, L. K. Balzano, and M. B. Srivastava. 2008. Reputation-based framework for high integrity sensor networks. *ACM Trans. Sen. Netw.* 4, 3 (2008), 1–37.

J. A. Goguen and J. Meseguer. 1982. Security policies and security models. In *IEEE Symposium on Security and Privacy*. 11–20.

R. Gorrieri and C. Versari. 2015. *Introduction to concurrency theory: Transition systems and CCS*. Springer.

J. Halpern and R. Pucella. 2012. Modeling adversaries in a logic for security protocol analysis. *Logical Methods in Computer Science* 8 (2012), 1–26.

G. Han, J. Jiang, L. Shu, J. Niu, and H.-C. Chao. 2014. Management and applications of trust in wireless sensor networks: A survey. *J. Comput. System Sci.* 80, 3 (2014), 602–617. Special Issue on Wireless Network Intrusion.

F. He, H. Zhang, H. Wang, M. Xu, and F. Yan. 2010. Chain of trust testing based on model checking. In *2nd Int. Conf. on Networks Security Wireless Communications and Trusted Computing (NSWCTC)*. IEEE, 273–276.

J. Hillston, J. Pitt, M. Wirsing, and F. Zambonelli. 2014. *Collective adaptive systems: Qualitative and quantitative modelling and analysis*. Technical Report. Dagstuhl Seminar 14512.

J. Huang and M.S. Fox. 2006. An ontology of trust – Formal semantics and transitivity. In *Proc. of 8th Electronic Commerce (ICEC'06)*. ACM.

D. Johnson, D. Maltz, and J. Broch. 2001. *Ad hoc networking*. Addison-Wesley, Chapter DSR: The dynamic source routing protocol for multihop wireless ad hoc networks, 139–172.

C. Jonker and J. Treur. 1999. Formal analysis of models for the dynamics of trust based on experiences. In *Procs. of 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW99) (LNCS)*, Vol. 1647. Springer, 221–232.

A. Jøsang. 2007. Trust and reputation systems. In *Foundations of Security Analysis and Design IV*. LNCS, Vol. 4677. Springer, 209–245.

S.-D. Kamvar, M.-T. Schlosser, and H. Garcia-Molina. 2003. The Eigentrust algorithm for reputation management in P2P networks. In *12th Conf. on World Wide Web (WWW)*. ACM, 640–651.

W.-S. Kim. 2009. Effects of a trust mechanism on complex adaptive supply networks: An agent-based social simulation study. *Journal of Artificial Societies and Social Simulation* 12, 3 (2009), 4.

M. Kwiatkowska, D. Parker, and A. Simaitis. 2013. Strategic analysis of trust models for user-centric networks. In *Int. Workshop on Strategic Reasoning (SR)*, Vol. 112. EPTCS, 53–60.

J. Li, R. Li, and J. Kato. 2008. Future trust management framework for mobile ad hoc networks. *IEEE Communications Magazine* 46, 4 (2008), 108–114.

Z. Li and H. Shen. 2012. Game-theoretic analysis of cooperation incentives strategies in mobile ad hoc networks. *Transactions on Mobile Computing* 11, 8 (2012), 1287–1303.

M. Loreti and J. Hillston. 2016. Modelling and analysis of collective adaptive systems. In *[Bernardo et al. 2016]*. 83–119.

L. Luisa Vissat, J. Hillston, G. Marion, and M. Smith. 2016. MELA: Modelling in ecology with location attributes. In *14th Workshop on Quantitative Aspects of Programming Languages and Systems (QAPL)*, Vol. 227. EPTCS, 82–97.

F. G. Marmol and G. M. Perez. 2009. Security threats scenarios in trust and reputation models for distributed systems. *Computers and Security* 28, 7 (2009), 545–556.

F. G. Marmol and G. M. Perez. 2011. Trust and reputation models comparison. *Internet Research* 21, 2 (2011), 138–153.

F. Martinelli. 2005. Towards an integrated formal analysis for security and trust. In *7th Conf. on Formal Methods for Open Object-Based Distributed Systems (FMOODS'05) (LNCS)*, Vol. 3535. Springer, 115–130.

F. Martinelli and M. Petrocchi. 2007. A uniform framework for security and trust modeling and analysis with crypto-CCS. In *First Workshop in Information and Computer Security*, Vol. 186. ENTCS, 85–99.

M. Momani. 2010. Trust models in wireless sensor networks: A survey. In *Recent Trends in Network Security and Applications: Third Int. Conference (CNSA)*. Springer, 37–46.

H. Mousa, S. Ben Mokhtar, O. Hasan, O. Younes, M. Hadhoud, and L. Brunie. 2015. Trust management and reputation systems in mobile participatory sensing applications: A survey. *Computer Networks* 90 (2015), 49–73.

T. Muller. 2010. Semantics of trust. In *7th Int. Workshop on Formal Aspects in Security and Trust (FAST'10) (LNCS)*, Vol. 6561. Springer, 141–156.

E. C. H. Ngai and M. R. Lyu. 2004. Trust- and clustering-based authentication services in mobile ad hoc networks. In *24th Int. Conf. on Distributed Computing Systems Workshops*. IEEE, 582–587.

D.M. Nicol and J. Huang. 2010. A Formal-semantics-based calculus of trust. *Internet Computing* 14, 5 (2010), 38–46.

H. S. Packer, L. Dragan, and L. Moreau. 2014. An auditable reputation service for collective adaptive systems. In *Social Collective Intelligence: Combining the Powers of Humans and Machines to Build a Smarter Society*. Springer, 159–184.

M. Reith, J. Niu, and W. H. Winsborough. 2007. Apply model checking to security analysis in trust management. In *IEEE 23rd Int. Conf. on Data Engineering Workshop*. IEEE, 734–743.

Y. Ben Saied, A. Olivereau, D. Zeghlache, and M. Laurent. 2013. Trust management system design for the Internet of Things: A context-aware and multi-service approach. *Computers & Security* 39, Part B (2013), 351–365.

A. Schlosser, M. Voss, and L. Brckner. 2004. Comparing and evaluating metrics for reputation systems by simulation. In *Procs. of IAT Workshop on Reputation in Agent Societies*.

A. Tarable, A. Nordio, E. Leonardi, and M. G. Ajmone Marsan. 2015. The importance of being earnest in crowdsourcing systems. In *IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2821–2829.

M. ter Beek, A. Fantechi, S. Gnesi, and F. Mazzanti. 2008. An action/state-based model-checking approach for the analysis of communication protocols for service-oriented applications. In *12th Workshop on Formal Methods for Industrial Critical Systems (FMICS)*, Vol. 4916. LNCS, 133–148.

G. Theodorakopoulos and J.-S. Baras. 2006. On trust models and trust evaluation metrics for ad hoc networks. *Journal on Selected Areas in Communications* 24 (2006), 318–328.

D. Trcek. 2009. A formal apparatus for modeling trust in computing environments. *Mathematical and Computer Modelling* 49, 1–2 (2009), 226–233.

R.J. van Glabbeek. 2011. On cool congruence formats for weak bisimulations. *Theoretical Computer Science* 412, 28 (2011), 3283–3302.

R. J. van Glabbeek and F. Vaandrager. 1987. Petri net models for algebraic theories of concurrency. In *Proc. of Parallel Languages (PARLE'87) (LNCS)*, Vol. 259. Springer, 224–242.

Y. Wang, Z. Cai, G. Yin, X. Tong, and Q. Han. 2016. A game theory-based trust measurement model for social networks. *Computational Social Networks* 3, 2 (2016).

L. Xiong and L. Liu. 2004. PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Trans. on Knowl. and Data Eng.* 16, 7 (2004), 843–857.

R. Yaich, O. Boissier, P. Jaillon, and G. Picard. 2012. An adaptive and socially-compliant trust management system for virtual communities. In *27th Annual ACM Symposium on Applied Computing (SAC)*. ACM, 2022–2028.

Y. Yu, K. Li, W. Zhoub, and P. Lib. 2012. Trust mechanisms in wireless sensor networks: Attack analysis and countermeasures. *Journal of Network and Computer Applications* 35, 3 (2012), 867–880.

Y. Zhang, L. Lin, and J. Huai. 2007. Balancing trust and incentive in peer-to-peer collaborative system. *Journal of Network Security* 5 (2007), 73–81.

Received January 2017; revised June 2017; accepted October 2017