UNIVERSITÀ
DEGLI STUDI
DI PALERMO

NDS
NETWORKING AND DISTRIBUTED SYSTEMS
RESEARCH GROUP

A Framework for Parallel Assessment of Reputation Management Systems

Article

Accepted version

V. Agate, A. De Paola, S. Gaglio, G. Lo Re, M. Morana

In Proceedings of the 17th International Conference on Computer Systems and Technologies

A Framework for Parallel Assessment of Reputation Management Systems

Vincenzo Agate, Alessandra De Paola, Salvatore Gaglio, Giuseppe Lo Re, Marco Morana

Abstract: *Several distributed applications running over the Internet use Reputation Management Systems (RMSs) to guarantee reliable interactions among unknown agents. Because of the heterogeneity of the existing RMSs, their assessment in terms of correctness and resistance to security attacks is not a trivial task. This work addresses this issue by presenting a novel parallel simulator aimed to support researchers in evaluating the performances of a RMS since the design phase. Preliminary results obtained by simulating two different attacks confirm the suitability of the proposed framework to evaluate different RMSs.*

Keywords: *Parallel Simulation, Distributed Reputation Management, Multi-threaded Applications.*

INTRODUCTION

In many distributed systems, the request for resources and services passes through unknown agents with unpredictable behaviours. This may lead malicious or selfish agents to cause a detriment of the performance of the whole system. One of the most effective way of addressing this issue is to rely on Reputation Management Systems to encourage agents to act honestly and cooperatively. In particular, in a fully distributed environment that lacks of a single centralized authority capable of coordinating all the interactions among agents, each member of the community may contribute in estimating the reputation of the agents so as to reward well-behaved ones.

The state-of-the-art models for reputation management show relevant differences due to the heterogeneity of the considered application scenarios, such as peer-to-peer applications for file sharing [16], e-Commerce frameworks [7] and service-oriented architectures [15][5]. Due to such diversity, there is the lack of a general tool to perform large-scale simulations for easily assessing new approaches with respect to their correctness and resistance to security attacks [19]. Then, the choice most frequently made by the researchers for evaluating RMSs is to design ad-hoc simulators. However, even if ad-hoc simulators allow to overcome the lack of real-world dataset, they are not suitable to compare different approaches. Some testbeds have been proposed in literature, but none of them allow to reach the goals stated above.

This work[1] presents a novel parallel simulator, preliminarily described in [1], aimed to support researchers in evaluating the performances of a general RMS. The system is organized in two logic layers that allow for separating the high-level representation, i.e. the reputation network, from the distributed processes that actually implement the behaviour of the agents involved in the simulation.

The remainder of the paper is organized as follows: first, related work is reported. Then, the RMS used for testing the performance of the simulator is described. The third and fourth sections provide a detailed description of the framework, and present a set of experiment aimed to verify its correctness in simulating two different attacks.

[1] Work partially supported by the Italian Ministry of Education, University and Research on the "StartUP" call funding the "BIGGER DATA" project PAC02L1 0086 CUP: B78F13000700008.

RELATED WORK

Several reputation management models for distributed systems have been proposed in the literature. The authors of [19] analysed the main components of trust systems in the context of Multi-Agent Systems (MASs), identifying a trust evaluation phase, which assesses the reliability of the agents involved in the interactions, and a trust-aware decision making phase that uses reputation values to select the agents to interact with.

RMSs for distributed systems, where a central authority is missing, belong to the second class, according to which each agent relies on a distributed protocol to obtain opinions from other agents, and merges them with its past experience in order to obtain the reputation of a given agent. In such a distributed scenario, security is a critical issue, since RMSs are susceptible to different type of attacks, as described in [11]. It is possible to identify five classes of attacks: *self-promoting*, *slandering*, *orchestrated*, *whitewashing*, and *denial of service*. In self-promoting attacks, malicious agents exploit system vulnerabilities in order to increment their own reputation, whilst in slandering their goal is to decrease the reputation of some "victim" agents. Both attacks may be performed according to an orchestrated plan that requires the coordination among several malicious agents. Whitewashing attacks aim to "clean" the bad reputation of a malicious agent to avoid the negative effects of the disincentive system. A denial-of-service attack aims to block the functioning of the system, i.e., to hinder a reliable reputation evaluation.

In the present work we focus on whitewashing and slandering attacks. The most common method adopted by malicious agents to perform a whitewashing attack is to exploit system resources until their reputation value goes under an acceptable threshold. Then, the agents leave the community and re-join it with a new identity associated with a default reputation value. RMSs are more vulnerable to whitewashing attacks when new agents have a default reputation value comparable to the long-term reputation of a honest agent, or when negative feedbacks are more relevant than positive ones. Moreover, a whitewashing attack can be reinforced by a combined self-promoting attack to extend its effect. Slandering attacks aim to change the reputation of other agents by disseminating false negative feedbacks in order to obtain some advantage. Single slanderers cause a limited effect, whilst a coordinated group of malicious agents may seriously damage the reputation of victim nodes. RMSs are more vulnerable to such attacks if gossiped opinions are more relevant than direct interactions. Moreover, the lack of a feedback authentication mechanism may intensify such vulnerability.

A popular simulation testbed in the field of multi-agent systems is ART (Agent Reputation and trust), proposed in [8]. ART allows to apply several evaluation metrics, and to define competitions in which different strategies can be combined and compared with respect to the utility obtained by each agent at the end of the simulation. Such feature, useful in MAS scenarios, is less relevant when the goal is to prove the capability of a RMS to discourage malicious behaviours in distributed systems. TREET [13] allows to evaluate RMSs in a marketplace scenario by measuring the resistance of the system with respect to some attacks (e.g., Reputation Lag Attack, Proliferation attack and Value Imbalance). TREET overcomes many limitations of ART, allowing agents to dynamically join or leave a simulation, even if the pattern of such events cannot be determined by experiments.

The authors of [1] propose a generic testbed for evaluating RMSs by modelling them as a sequence of transformations of a graph that represents transactions and trust among agents. Even if this testbed allows to evaluate security resistance to slandering and self-promotion attacks, it does not simulate agent interactions that are crucial to perform large-scale simulations where agents may modify their behaviour.

REPUTATION MANAGEMENT SYSTEM

The main purpose of a RMS is to detect and discourage antisocial behaviours that negatively affect the whole community. The role of a RMS is crucial in totally distributed

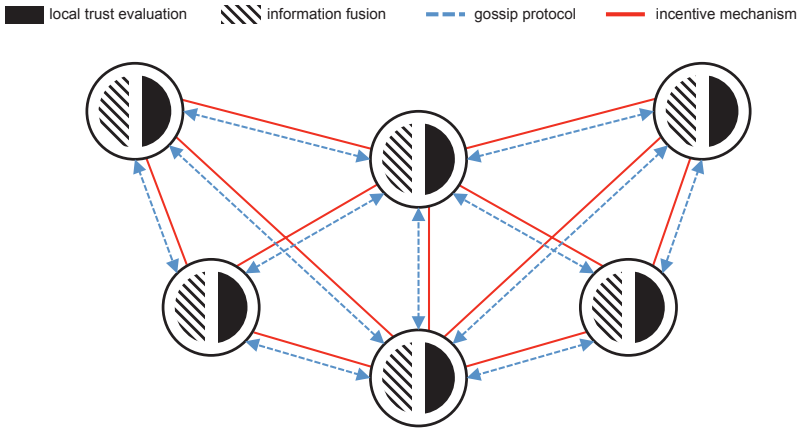

Fig. 1: A distributed RMS. Each agent privately performs local trust evaluation and information fusion algorithms; gossip protocol and incentive mechanism rule the interactions with other agents.

scenarios, where a centralized entity capable of coordinating the interactions among agents is missing. In order to design a generic framework, we identified the components common to many of the RMSs for distributed environments proposed in the literature (Fig. 1): (i) a local trust evaluation mechanism, used for assessing the behaviour of the agents involved in direct interactions, (ii) a gossip protocol, which propagates the local trust to other agents of the network, (iii) an information fusion mechanism, to merge information gathered through the gossip protocol with the local trust, and obtain the reputation values, and (iv) a disincentive mechanism which exploits reputation values in order to discourage antisocial behaviours. In this work we consider as case study to prove the effectiveness of our framework in the evaluation of RMSs' security resistance, a RMS that includes all these components and is inspired by [12] and [6].

The local trust evaluation mechanism is a variation of EigenTrust [12], one of the most known RMSs for P2P networks. Each agent *i* stores the number of satisfactory, *sat(i,j)*, and unsatisfactory, *unsat(i,j)*, transactions occurred with other agents *j* in the network. The local trust s_{ij} is defined as the difference between such values, i.e., s_{ij} = *sat(i,j)−unsat(i,j)*. In order to support the later information fusion phase, it is required to normalize such local trust values. We propose a variation of the normalization technique used in EigenTrust, characterized by some drawbacks highlighted by the same authors. Thus, our normalized local trust is obtained by scaling s_{ij} with respect to its maximum observed value and by cutting off negative values, as follows:

$$c_{ij} = \frac{max(s_{ij}, 0)}{max_j\{s_{ij}\}} \quad (1)$$

The gossip protocol, performed at each time step, states that each agent sends its reputation values to all its neighbours. At the end of this phase, each agent knows the opinion of all its one-hop neighbours about the reputation of its two-hop neighbourhood.

Reputation information gathered so far is then merged during the information fusion phase, inspired by [6]. Here each agent merges only information coming from reliable agents, i.e., those whose reputation is beyond a given threshold τ. Merged information is weighted with the reputation of the "gossiper" agents, and the resulting reputation value r_{ij} is a linear combination of this weighted mean and the local trust:

$$r_{ij}(t) = (1-\beta) * c_{ij}(t) + \beta * \frac{\sum_{k \in K} r_{ik}(t-1) * r_{kj}(t-1)}{\sum_{k \in K} r_{ik}(t-1)} \quad (2)$$

where β is a coefficient in [0,1] and K is the set of reliable agents:

$$K = \{k : r_{ik}(t-1) \geq \tau\}. \quad (3)$$

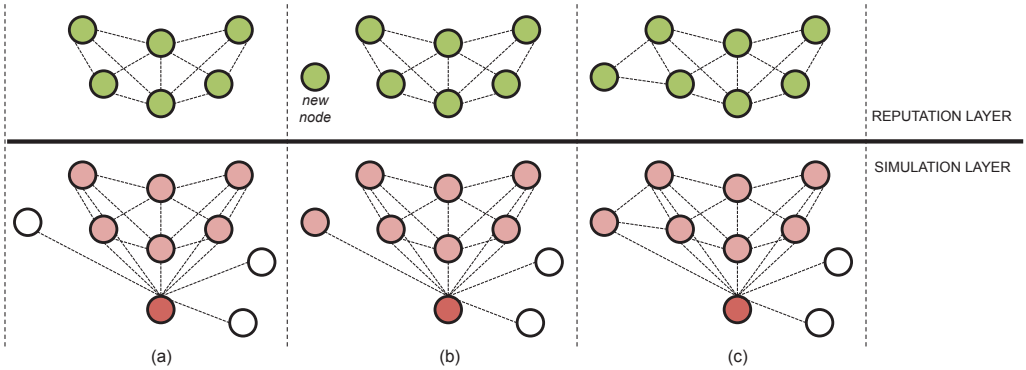

Fig. 2: Creation of a node as seen at the reputation and simulation layer. The agents (green) are modelled by active processess (light red) managed by a controller (dark red). Inactive processes are depicted as white circles.

The adoption of a weighted sum between the local trust value and the average values reported by other agents is a common solution in the literature [17][18].

In order to evaluate the effect of the RMS on agent behaviours, we adopt a disincentive mechanism that allows an agent to obtain resources with a probability proportional to its reputation, as described in [5]. Such solution is one of the most popular approaches to implement the disincentive system [19].

PARALLEL SIMULATOR DESIGN

The simulation framework we propose here is organized in two different logic layers (Fig. **2**) so as to separate low-level functionalities from the routines that drive the user in the simulation tasks, similarly to a previous work where we addressed the design of a simulator for Wireless Sensor Networks in a distributed scenario [14].

The topmost is the reputation layer, made of nodes connected to each other according to a specific network topology. At this level of abstraction, the reputation network is shown as totally distributed, non-centralized, and a set of high-level configuration utilities are offered to the user.

At the simulation layer, each agent is mapped to a different process. Here, a leading process called controller is responsible for creating new processes and change the behaviour of those that already exist. To be more specific, when the simulator is launched, a set of inactive processes is created (Fig. **2**-a). Then, as new nodes are added to the network, inactive processes are awoken by the controller to become active (Fig. **2**-b), and connected to the other processes involved in the simulation (Fig. **2**-c). The role of the controller can be further explained by considering a whitewashing attack scenario, where a node A (at the reputation layer) with a low reputation value wants to duplicate itself to keep exploiting the system resources. In such a situation, the corresponding active process PA sends a duplication request to the controller PC. Then, PC awakes and initializes one of the inactive processes with the characteristics, i.e. adjacency list and behaviour, of PA.

The simulation framework offers a number of routines that allow users to define the RMS to be evaluated. In particular, researchers can define the local trust evaluation technique, the gossip protocol, the information fusion algorithm, and the disincentive mechanism that each agent can exhibit.

In order to define the local trust, the simulation framework provides the number of requests that each agent has sent to other agents, together with the number of negative and positive feedback obtained by them. These values are updated at each time step; thus, the user can choose the most proper information granularity, e.g., all values obtained since the beginning of the simulation or the average value computed in a sliding window.

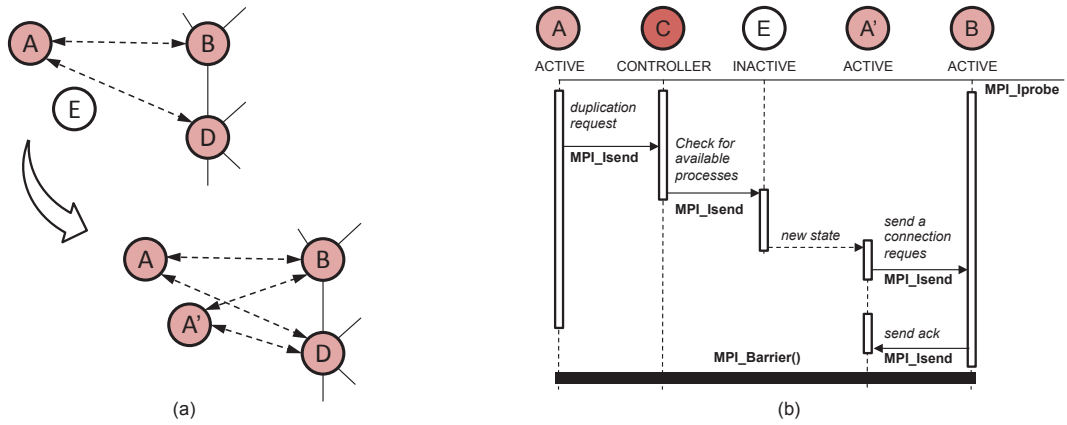

Fig. 3: Duplication of a node A. (a) The inactive process E is used by the controller to duplicate the process A; then A' is linked to B and D. (b) Interactions and MPI calls needed to duplicate the process A.

The gossip protocol can be defined by means of a set of routines that allow for obtaining information about the reputation network and for supporting communication among agents. In particular, it is possible to get the list of the current neighbours, send unicast messages to specific neighbours, and send broadcast messages to the whole neighbourhood.

The simulator manages the synchronization steps required to guarantee that each agent receives all messages sent by its neighbours, before performing the information fusion phase. Obviously, it is possible to specify which piece of information has to be communicated, e.g., the local trust or the global reputation, and how it can be merged in the information fusion phase. The disincentive mechanism is based on the trust and reputation values, and specifies the rules each agent must follow to reply to a resource request.

Different attacks can be performed by defining the behaviour of the agents involved in the simulation through a set of configuration files. Information to be specified includes the number of nodes involved in simulation, the cooperativeness degree of each agent, and the topology of the reputation network.

Furthermore, it is possible to specify the set of attacks to be simulated, by detailing, for each one, the starting time step and the list of the nodes involved in it. For a slandering attack, it is required to select a victim node V, and a set of M malicious nodes, adjacent to V, programmed to share with their neighbours a fake reputation value of the node V.
In order to simulate a whitewashing attack, it is necessary to specify the set of selfish nodes and the threshold on the obtained resources needed to trigger the duplication of a false identity. We also suggest to include neutral node in every type of simulation in order to track the simulation results, e.g., the reputation values of the victim.

From a logical point of view, each process can run on a distinct remote host, thus we adopted the Message Passing Interface (MPI) to enable communication among different distributed processes. MPI provides a protocol for parallel message-passing in distributed scenarios where processes exchange data through cooperative operations. The synchronization is managed by means of blocking and non-blocking, point-to-point or collective, communication primitives that guarantee safe access to shared data.

Some of the MPI primitives used by the simulator are shown in Fig. **3**, where the framework simulates a whitewashing attack by duplicating a node A with a low-reputation value. All processes use the thread-safe **MPI_Iprobe()** routine to manage incoming messages. Messages are sent and received by means of the non-blocking functions **MPI_Isend()** and **MPI_Irecv()** respectively, whilst **MPI_Barrier()** is used to synchronize all processes within the MPI communicator.

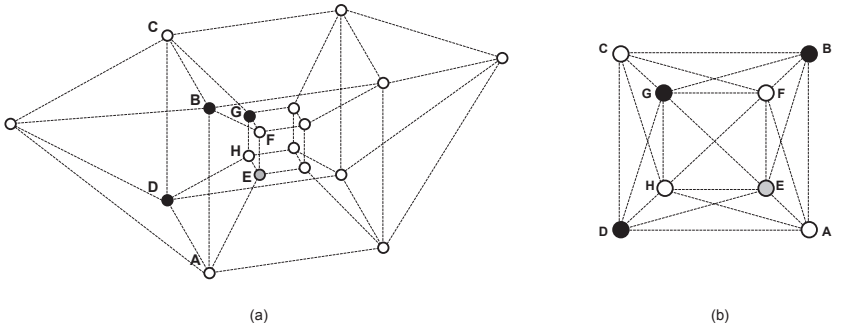

Fig. 4: (a) The network topology used for simulating the slandering and whitewashing attacks to the RMS. (b) A branch of the network showing the connections between normal (white), malicious (black) and victim (grey) nodes.

EXPERIMENTAL EVALUATION

The aim of the tests we performed is to evaluate the capability of the framework to simulate common attacks to a RMS. In particular, we present the assessment of the simulation framework addressing a scenario where slandering and whitewashing attacks are launched against the RMS described in Section 3.

At the end of the simulation, the framework returns the reputation value and the percentage of obtained resources for all agents involved in the simulation, for each time step. The comparison of the estimated reputation value for a given agent with its cooperativeness degree, set through a simulation parameter, allows to evaluate the average accuracy rate of a RMS, which is one of the most adopted metrics by works presented in the literature [19], together with the average utility that can be directly computed from the percentage of obtained resources. As compared to aggregated, average, results, the detailed trends of reputation and obtained resources over time allow to perform deeper and more various analyses of behaviour of the observed RMS.

The network topology adopted for evaluating the effect of a slandering attack is shown in Fig. **4**-a, whilst Fig. **4**-b shows the connections between normal (white) and malicious (black) nodes adjacent to a victim (grey) node. During the whole simulation the victim agent E acts honestly, but its reputation is negatively affected by false information disseminated by malicious agents D, G, and B in the reputation gossip phase. Such effect can be observed by focusing on the top-left part of Fig. **5**, which shows the reputation of E as seen by the agent A, that receives information about E from its neighbours D, H, F, B (see Fig. **4**-b).

The consequences of the attack are not instantaneous due both to the positive effect of direct interactions, and to the past history. In order to compare the vulnerability of different RMSs to slandering attacks, it is possible to analyse the time necessary to waste the reputation of a honest agent and the required percentage of malicious gossipers. The bottom-left part of Fig. **5** shows that the percentage of resources obtained by the victim agent decreases proportionally to its reputation.

The same analysis can be conducted by considering a whitewashing attack where a malicious agent D acts selfishly with respect to the whole community, by responding only to 10% of the received requests. The top-right part of Fig. **5** shows that the reputation of D, as seen by its honest neighbour A, decreases over time. When the reputation value goes under a fixed threshold, i.e., after 80 time steps, the malicious agent D duplicates itself by creating a new identity with a default reputation value. The bottom-right part of Fig. **5** shows the percentage of resources obtained by D using both its identities. In order to compare two RMSs with respect to their resistance to a whitewashing attack, we suggest to compare the whole amount of resources obtained by malicious agents by considering all their false identities.

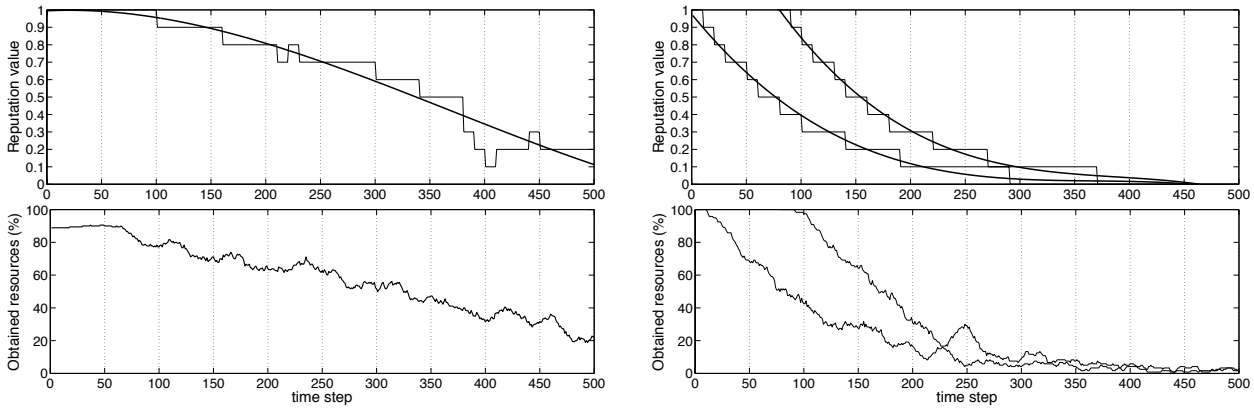

Fig. 5: Reputation values (top) and corresponding percentage of resources (bottom) obtained by a victim agent during a slandering attack (left) and by a malicious agent during a whitewashing attack (right).

CONCLUSIONS AND FUTURE WORK

In this work we presented a simulation framework aimed to support researchers in evaluating the performances of a Reputation Management System since the design phase. Such a tool is based on a two-layer architecture that separates reputation management techniques from the software routines that actually drive the simulation. Each agent of the reputation layer is mapped to a process running at the simulation layer, where MPI interfaces are used to enable communication among different distributed processes.

The framework allows the user to specify the topology of the simulation network and all the parameters which define the behaviour of the simulated agents. The simulator has been tested by conducting an experimental campaign aimed to verify its correctness in simulating two different attacks against a RMS, namely slandering and whitewashing attacks.

The results we obtained demonstrate the suitability of the proposed framework for providing detailed information required to conduct deep analyses about the performance of the considered RMS.

As future work we want to provide an improved graphical interface allowing users to easily change the simulation parameters, and to personalize the framework by defining custom agent behaviours. Moreover, we are considering the adoption of the simulation framework to mimic the interactions within large-scale social networks, e.g., Twitter [10], [9] and to model the user's behaviour [3], [4].

REFERENCES

[1] V. Agate, A. De Paola, G. Lo Re, M. Morana. A Simulation Framework for Evaluating Distributed Reputation Management Systems. In: Proc of the 13th Int. Conf. on Distributed Computing and Artificial Intelligence. pp 247-254 (2016)

[2] Chandrasekaran, P., Esfandiari, B.: Toward a testbed for evaluating computational trust models: experiments and analysis. Journal of Trust Management 2(1), 1–27 (2015)

[3] P. Cottone, S. Gaglio, G. Lo Re, M. Ortolani. A Machine Learning Approach for User Localization Exploiting Connectivity Data. In Journal of Engineering Applications of Artificial Intelligence (Elsevier)

[4] P. Cottone, S. Gaglio, G. Lo Re, M. Ortolani. User Activity Recognition for Energy Saving in Smart Homes. In Proc. of the 3rd Int. Conf. on Sustainable Internet and ICT for Sustainability, 2013, pp. 1-9

[5] Crapanzano, C., Milazzo, F., De Paola, A., Lo Re, G.: Reputation management for distributed service-oriented architectures. In: Proc. of the 4th IEEE Int. Conf. on Self-Adaptive and Self-Organizing Systems Workshop. pp. 160–165. IEEE (2010)

[6] De Paola, A., Tamburo, A.: Reputation Management in Distributed Systems. In: Proc of the 3rd Int. Symposium on Communications, Control and Signal Processing (ISCCSP). pp. 666–670 (2008)
[7] Fouliras, P.: A novel reputation-based model for e-commerce. Operational research 13(1), 113–138 (2013)
[8] Fullam, K.K., Klos, T.B., Muller, G., Sabater, J., Schlosser, A., Topol, Z., Barber, K.S., Rosenschein, J.S., Vercouter, L., Voss, M.: A specification of the agent reputation and trust (ART) testbed: experimentation and competition for trust in agent societies. In: Proc. of the 4th Int. joint Conf. on Autonomous agents and multiagent systems. pp. 512–518. ACM (2005)
[9] Gaglio, S., Lo Re, G., Morana, M.: Real-time detection of Twitter social events from the user's perspective. In: Communications (ICC), 2015 IEEE Int. Conf. on. pp. 1207–1212 (2015)
[10] Gaglio, S., Lo Re, G., Morana, M.: A framework for real-time Twitter data analysis. Computer Communications 73, Part B, 236 – 242 (2016)
[11] Hoffman, K., Zage, D., Nita-Rotaru, C.: A survey of attack and defence techniques for reputation systems. ACM Computing Surveys (CSUR) 42(1), 1 (2009)
[12] Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The Eigentrust algorithm for reputation management in p2p networks. In: Proc. of the 12th Int. Conf. on World Wide Web. pp. 640–651. ACM (2003)
[13] Kerr, R., Cohen, R.: Smart cheaters do prosper: defeating trust and reputation systems. In: Proc. of The 8th Int. Conf. on Autonomous Agents and Multiagent Systems-Volume 2. pp. 993–1000. (2009)
[14] Lalomia, A., Lo Re, G., Ortolani, M.: A hybrid framework for soft real-time WSN simulation. In: Distributed Simulation and Real Time Applications, 2009. DS-RT '09. 13th IEEE/ACM Int. Symposium on. pp. 201–207 (2009)
[15] Lee, J., Zhang, J., Huang, Z., Lin, K.J.: Context-based reputation management for service composition and reconfiguration. In: Proc. of the 2012 IEEE 14th Int. Conf. on Commerce and Enterprise Computing (CEC). pp. 57– 61 (2012)
[16] Marti, S., Garcia-Molina, H.: Taxonomy of trust: Categorizing P2P reputation systems. Computer Networks 50(4), 472–484 (2006)
[17] Weng, J., Shen, Z., Miao, C., Goh, A., Leung, C.: Credibility: How agents can handle unfair third-party testimonies in computational trust models. IEEE Trans- actions on Knowledge and Data Engineering 22(9), 1286–1298 (2010)
[18] Yu, H., Liu, S., Kot, A., Miao, C., Leung, C.: Dynamic witness selection for trust-worthy distributed cooperative sensing in cognitive radio networks. In: Proc. of the 2011 IEEE 13th Int. Conf. on Communication Technology (ICCT). pp. 1–6 (2011)
[19] Yu, H., Shen, Z., Leung, C., Miao, C., Lesser, V.R.: A survey of multi-agent trust management systems. Access, IEEE 1, 35–50 (2013)

ABOUT THE AUTHORS

Vincenzo Agate, vincenzo.agate@unipa.it, University of Palermo, Italy
Assist. Prof. Alessandra De Paola, alessandra.depaola@unipa.it, University of Palermo, Italy
Full Prof., Salvatore Gaglio, salvatore.gaglio@unipa.it, University of Palermo, Italy
Assoc. Prof. Giuseppe Lo Re, giuseppe.lore@unipa.it, University of Palermo, Italy
Assist. Prof. Marco Morana, marco.morana@unipa.it, University of Palermo, Italy