# An Experimental Testbed for Evaluation of Trust and Reputation Systems

Reid Kerr and Robin Cohen

David R. Cheriton School of Computer Science,
University of Waterloo, Waterloo, Ontario, Canada
{rckerr,rcohen}@uwaterloo.ca

**Abstract.** To date, trust and reputation systems have often been evaluated using methods of their designers' own devising. Recently, we demonstrated that a number of noteworthy trust and reputation systems could be readily defeated, revealing limitations in their original evaluations. Efforts in the trust and reputation community to develop a testbed have yielded a successful competition platform, ART. This testbed, however, is less suited to general experimentation and evaluation of individual trust and reputation technologies. In this paper, we propose an experimentation and evaluation testbed based directly on that used in our investigations into security vulnerabilities in trust and reputation systems for marketplaces. We demonstrate the advantages of this design, towards the development of more thorough, objective evaluations of trust and reputation systems.

## 1 Introduction

The area of *multiagent systems* is concerned with scenarios where a number of agents (who may be acting on behalf of different users) must interact in order to achieve their goals; often, an agent must depend on other agents in order to achieve its objectives. In such scenarios, trust can be an important issue—an agent's ultimate success may depend on its ability to choose trustworthy agents with which to work. For this reason, *trust and reputation systems* (TRSes)[1] have received much attention from researchers. Such systems seek to aid agents in selecting dependable partners (or in avoiding undependable ones).

A particular focus for researchers has been on the electronic marketplace scenario, a well-established and important example of a multiagent system. In this setting, agents act as traders, buying and selling amongst one another. The ability to find trustworthy partners is critical to an agent's success, because an untrustworthy agent may deliver an inferior good (or fail to deliver at all), or may not pay for goods purchased. The nature of electronic marketplaces complicates the evaluation of trustworthiness: identity is difficult to establish (because new accounts can be created easily), agents may not engage in repeated transactions

[1] For convenience, we use the abbreviation TRS, for 'Trust/Reputation System', in reference to both trust systems and reputation systems.

E. Ferrari et al. (Eds.): TM 2009, IFIP AICT 300, pp. 252–266, 2009.

together (because of the size of the market and the diversity of products), and an agent may have an advantage over another during a transaction (for example, when a buyer must pay in full before a seller ships (or fails to ship) the good).

Along with the multitude of TRS proposals have come a similarly large number of methods to evaluate the proposals. It has been widespread practice for researchers in the field to develop their own testing methods. A common approach has been to conduct simulations using a scenario of the authors' own devising to show the value of their model, often achieved by pitting their new proposal against their implementations of other existing models. There is nothing fundamentally unreasonable about this approach, in the absence of established testing tools. Unfortunately, there have been significant limitations in the evaluations typically used by authors. It is not surprising that testing scenarios used by authors often favour their own work. This is not to suggest any misdeeds on the part of these authors; when designing a system, it is natural to have a particular scenario in mind, and for subsequent tests to reflect that scenario.

Members of the trust and reputation community have invested significant effort in developing the Agent Reputation and Trust (ART) testbed [1]. A primary purpose of ART is to serve as a competition platform, and it serves this purpose well [2]. While ART is a valuable contribution, a number of design choices make it less appropriate for broad use in the experimental evaluation of TRSes. We discuss these issues in Sect. 2.1.

Perhaps more important than issues of bias, the evaluation procedures typically used obscure critical problems that have received insufficient attention to date by trust and reputation researchers. Simulations typically make use of agents that are simple or naive in their dishonest activities. For example, many simulations (e.g., [3,4]) are populated by random selections of agents that either always cheat or always behave honestly, or by agents whose cheating is governed by simple probability distributions, where each time step is independent of previous ones. Such simulations ignore the possibility that cheaters might behave in a more sophisticated manner—for example, trying to identify and exploit a specific weakness in the system—providing little comfort to those who might wish to consider these proposals for real-world use. In earlier work [5], we identified a number of common vulnerabilities that might allow attackers to defeat the protection offered by TRSes, and argued the critical importance of security in TRSes. Recently [6], we demonstrated the practicality of such attacks by soundly defeating a number of noteworthy TRS proposals. These results demonstrate the need for more rigorous tests, and more objective tests, of TRSes.

In this paper, we propose a testbed formulation designed to support diverse, flexible experimentation with TRSes, and more thorough, objective evaluation of such systems. This formulation is based directly on the platform used in our experimentation in [6], which was designed to be a general-purpose experimental platform for both of these purposes. Our experience and results have shown it to strongly support both goals.

The design has a number of important advantages making it well suited for its intended purposes, including:

- It models a general marketplace scenario, allowing systems to be tested under realistic conditions. This includes reasonably large marketplace populations, turnover in the agent population, a large number of products/prices, etc.
- It is modular, allowing new TRSes, buying and selling agents, instrumentation, etc., to be added easily.
- It can support a wide range of trust/reputation approaches (for example, both centralized and decentralized models). It does not impose any particular view of trust on agents, nor does it impose a particular protocol or trust representation on agents.
- It allows collusion to be incorporated into agent behaviour.
- It allows individual marketplace 'components' to be tested in isolation. For example, it allows the protection a TRS provides buyers from cheating sellers to be evaluated, without being obscured by other potentially irrelevant issues (for example, whether or not sellers are dishonest with one another). In contrast, the success of an agent in ART requires competence in a number of abilities.
- Given the standardized platform, as new agents/TRSes are developed, they can be evaluated against all existing implementations; at the same time, new implementations constitute new tests for all of the existing systems. In this way, a continually improving battery of rigorous tests can be developed, which can be broadly used by researchers to evaluate their work. (The 'smart' cheating agents used in [6], for example, constitute an initial set of tests that have proven extremely difficult for existing proposals.)
- Standardization also allows for objective benchmarking, permitting meaningful comparison between systems. Moreover, the availability of components allows for results to be reproduced by other investigators.

We believe this testbed to be a valuable tool in its own right; moreover, we believe it to be an important step towards thorough, objective evaluation of trust and reputation systems.

## 2 Related Work

The majority of TRS proposals applicable to marketplaces have been evaluated using methods of their authors' own devising. (Subsequently, these systems might appear as comparison data points in later proposals' own self-devised tests.) Methods used have included mathematical analysis of properties (e.g., [5,7]) and simulation (e.g., [3,4,8]). Both of these approaches are reasonable. The evaluations performed, however, have proven to be problematic. Because each evaluation is different, results presented by different authors are not comparable. The evaluations presented are often quite brief, leading one to question whether the results thoroughly reveal the performance of the systems in question. Indeed, our investigations [6] revealed numerous ways in which the systems cited above can be defeated—issues that were not revealed in the authors' own analyses. These issues highlight the need for more thorough, objective testing of TRSes, ideally using tools that allow comparison and reproducibility of test results.

### 2.1 The ART Testbed

A standardized, common testing platform can potentially address the issues noted above. The Agent Reputation and Trust (ART) Testbed [1] is the only well-known trust and reputation testbed of which we are aware. ART has been well supported by the community, and has been used as a competition testbed at a number of conferences.

In ART, agents are art experts, each with varying levels of expertise in different eras. Agents are periodically asked to appraise pieces of art by clients. The accuracy of the appraisals given to clients determines how much business each agent will receive in the future, according to a fixed mechanism used by the testbed. The agent can choose how much to invest in generating its appraisal—greater investment yields greater accuracy. If an agent is asked to evaluate a piece from an era about which he is not knowledgeable, he can seek appraisals from other agents. Agents can also share information with one another about the reliability of other agents' appraisals.

ART is very well-designed for its primary purpose: evaluating agents (who use a number of abilities) in a competitive manner, using a small social trust scenario. ART has a number of desirable properties for a testbed. It offers a well-specified, standardized testing scenario and set of rules. It allows new agents to be easily implemented and plugged into the system; agents can then be used by others for future experimentation. It provides objective metrics for comparison between systems. That said, ART has a number of features that make it less suited for general-purpose trust and reputation experimentation. Other authors (e.g., [9,10]) have noted obstacles to using ART for evaluating their own work.

Under ART, the distinction between buying and selling agents is unclear, making some forms of experimentation problematic. The ultimate purchasers of appraisals (the 'clients') are buyers, and as such, agents serve as sellers for these transactions. Note, however, that clients' method of choosing appraisers (based on past performance) is fixed by the ART specification, precluding experimentation with buyer-side modelling of sellers for these transactions; similarly, it obviates investigation of sellers modelling potentially unreliable buyers. In contrast, as agents buy and sell appraisals with one another, each agent acts as both buyer and seller. Success under these circumstances requires skill in several areas: determining when to make do with your own knowledge, and when to seek help; determining how much to invest in appraisals; determining whom to trust when seeking appraisals; and determining whether or not to be honest when another agent asks you for help. While this is a demanding test, and one appropriate for a competition testbed, it can also obscure the role of each individual skill in an agent's performance. This makes it difficult to isolate individual marketplace components for evaluation. For example, if a researcher wishes to evaluate the performance of a system intended to allow sellers to model untrustworthy buyers, it may not useful to have the results clouded by the same agent's performance in the unrelated task of deciding whether or not to make honest sales to other agents.

In its role as a competition testbed, ART requires a very well-defined scenario. Unfortunately, this requirement seems to limit the flexibility of the system for experimentation. A number of design choices limit the range of investigations that can be performed using ART. For example, the ART architecture allows decentralized and direct experience models, but precludes testing of centralized models, because the method of sharing information amongst agents is specified by the testbed. It also prevents experimentation with models that regulate an entire marketplace (e.g., mechanism-design based approaches). Features of the chosen scenario prevent investigation of important issues. For example, each appraisal has a fixed price under ART, preventing exploration of vulnerabilities such as Value Imbalance (where a seller builds reputation by honestly executing small-value sales, then uses the reputation gained to cheat on larger ones [5]). The quality of an agent's appraisal is reflected in clients' decisions in the next timestep, preventing exploration of vulnerabilities such as Reputation Lag (where a seller can cheat a large number of sellers for a period of time before his reputation is updated to warn other potential victims [5]).

ART provides a heterogeneous environment where agents share reputation information with agents using other trust and reputation models. To permit communication between agents with different internal models, the format of communication is determined by the ART specification. This imposes a specific trust representation for communication between agents (if not for agents' internal use); the imposed format may not map well to the TRS's native representation, potentially disadvantaging the TRS.

Seeking to clarify why ART is not well-suited for some forms of experimentation, we have focused on a number of it limitations. After so doing, we wish to reiterate that ART is an excellent competition testbed for decentralized social trust and reputation models.

In contrast to the competition focus of ART, the testbed formulation we propose is designed specifically to support general-purpose experimentation and evaluation of trust and reputation technologies. Our proposal is based directly on the platform used in our study of the security of TRS proposals [6]; pertinent aspects of this study are discussed later in the paper.

## 3 Testbed

We sought to formulate a testbed that would support flexible experimentation and meaningful evaluation of trust and reputation technologies. Complete marketplaces may have many TRS components, from a range of possibilities: agents who have individual (and heterogeneous) internal models of other agents' trustworthiness, networks of agents that share reputation information, centralized repositories of reputation data, market-wide mechanisms that regulate trading between agents, etc. A potential adopter of a TRS may have to choose between multiple proposals, despite the fact that the proposals use very different methods internally. An adopter may have to assemble multiple TRS technologies to meet the needs of their complete working system, and may need to understand how

well these components work together. For these (and other) reasons, a testbed will ideally support experimentation with a wide variety of such components. Thus, we set out to design an architecture that was quite flexible.

At the same time, too general a testbed formulation might also be difficult to apply in practical terms. At best, it may be of little benefit to the researcher, leaving much work to be done simply in preparing the testing platform. Worse, a formulation that is too general can make evaluation of TRSes and comparison of results problematic: different researchers are likely to use very different instantiations of the testbed scenario, raising many of the same issues as the author-devised testing that has occurred to date. For this reason, we have specified a well-defined scenario that we believe is useful for a wide range of experimentation. We believe that this is an appropriate and useful balance between flexibility and standardization.

### 3.1 Nature of Tests

For a competition testbed, it is sufficient to supply the testing platform itself; competitors supply the agents, which seek to defeat one another. In contrast, a testbed intended for evaluation and benchmarking requires meaningful tests for candidates to perform. In some fields (for example, computer component benchmarking), a typical approach would be to develop a set of standardized tasks to perform, with well-defined metrics used for comparison. Ideally, the tasks would be representative of real-world demands. For TRSes, however, it is difficult to envision representative 'tasks' that do not involve actual interaction with other agents. The most illuminating tests are likely to be those conducted in a realistic scenario, interacting with other agents. Thus, in our formulation, tests consist of two components (in addition to the TRS technology being evaluated): a well-defined marketplace scenario, and a population of agents with which the candidate TRS must cope.

This formulation provides a great deal of flexibility, as well as the ability to test specific components under controlled circumstances. For example, to test TRSes that attempt to allow buyers to cope with cheating sellers, a test would consist of a set of market parameters, and a population of sellers with specific cheating behaviours. These components are experimental controls; each TRS would then be tested against the same scenario, allowing comparison of the results. (This was the approach used in our study [6].) In comparison, to test TRSes that allow sellers to cope with untrustworthy buyers, a test would include a set of buying agents.

Beyond the benefits noted above, this approach has a number of advantages. First, as agents are developed (both TRS technologies, and 'tests'), they can be made available to other researchers. This allows the test suite to grow, increasing in thoroughness and rigour, as understanding of TRSes increases. (Our set of cheating agents constitutes an initial set of tests, as outlined in a later section.) Second, the standardization of the platform and the availability of agents allows results to be reproduced by other researchers.

### 3.2 Scenario

We sought to develop a testbed that employs a reasonably general scenario, one in which a variety of roles and strategies can be evaluated. For tests to be meaningful, the platform should model as realistic a scenario as practically possible. In the following, the parenthesized parameter values represent settings for a reasonable scenario, one that might constitute a basis for test sets. (These values were used in our own experimentation [6].) While a test specification would include a set of parameter values, the values can be adjusted for experimentation.

We model an 'advertised-price' marketplace: sellers offer goods for sale, and buyers choose whether or not to make purchases, and from whom. A fixed set of products (1000) is available for sale. Because we wish to study trust primarily, and not other price-/cost-based forms of competition, the cost to produce/acquire any given good is the same for all sellers. A typical marketplace will have more inexpensive items for sale than expensive ones. To reflect this, the cost of each good is randomly determined using the right half of a Gaussian distribution (i.e., the median occurs at $0, and probability decreases as price increases). Again, to remove focus from price-based competition, all sellers apply a fixed markup (25% of selling price)—for a given good, all vendors charge the same price.

Each seller is assigned a random number of products that she is able to produce, selected from a uniform distribution (maximum of 10). To reflect the greater availability of less expensive products, the products are again randomly assigned using the right half of a Gaussian distribution (i.e., the median occurs at the least expensive product, with declining probability as price increases).

A simulation run can be populated by an assortment of agents, as desired by the researcher, or as defined in a test specification.

Marketplaces are usually dynamic—traders join and leave regularly. This is important for TRSes, because new agents are unknown (and have no knowledge of other agents), and departing agents result in obsolete knowledge. For efficiency, agents join/exit the market at specific intervals (100 days). After each such interval, each agent departs the marketplace with a fixed probability (0.05). That said, it may be undesirable for the performance of TRSes to be clouded by changes in market size (e.g., profits increasing because the number of buyers increases.) Thus, for every departing agent, one agent of the same type joins, keeping the participant count constant.

### 3.3 Architecture

The testbed architecture is designed to be quite versatile for experimentation, within the constraints of the defined scenario. The architecture is depicted in Fig. 1. In this diagram, BA and SA refer to Buying Account and Selling Account respectively. BE and SE refer to Buying Entity and Selling Entity respectively. All components labelled in boldface italic text are components that are intended to be provided/modified by investigators making use of the testbed. The grey box denotes those components that are observable by marketplace participants,

although this does not imply complete visibility. For example, seller accounts may be visible to buyer accounts, but this does not imply that all seller account data is visible. Such limitations are described in more detail below.

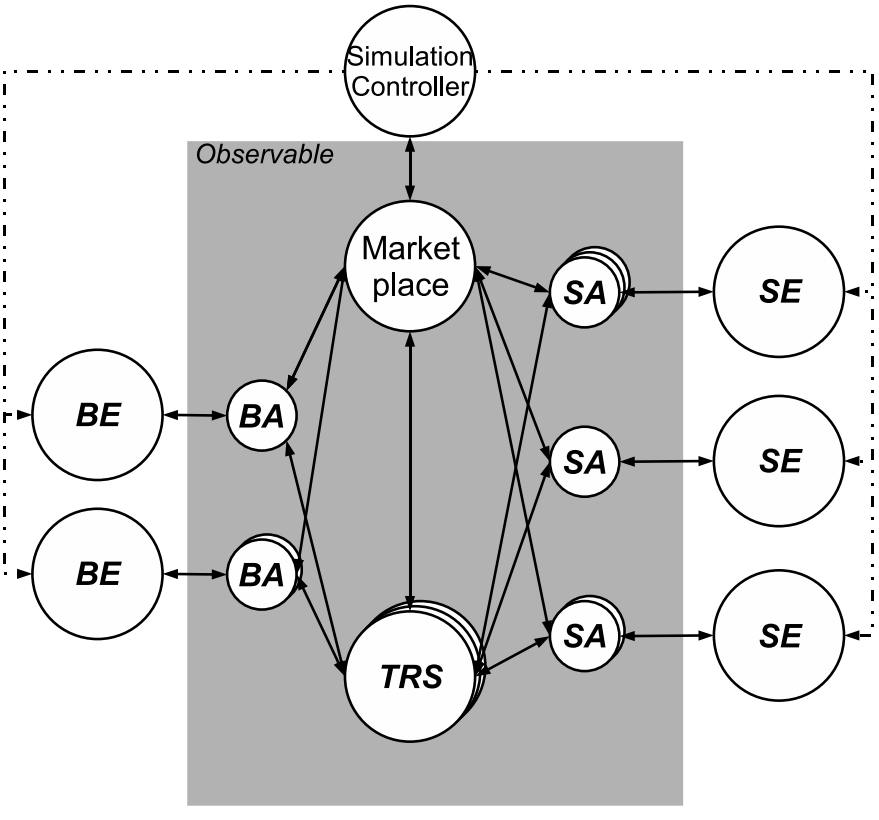


**Fig. 1.** The Testbed Architecture

A *Simulation Controller* is responsible for actual execution of the simulation. The controller is responsible for triggering each of the day's events in turn, signaling the appropriate parties when they are required to take action. For example, the controller cues sellers to make product offers at the appropriate times, cues buyers to select products/sellers when offers have been posted, etc.

The scenario makes use of a single centralized marketplace model, represented by a *Marketplace* object. All offers, acceptances, and payments are made through the Marketplace. All accounts reside in the Marketplace, and requests to open accounts are processed through it.

One important aspect of the testbed is the role of TRSes and agents. Some TRSes are implemented entirely centrally, some entirely within the agents; many fall between these two extremes. The use of both agents (represented by accounts

and entities, as described below), and TRS objects, facilitates a wide range of approaches. A *TRS* object implements those components of a TRS that are shared by multiple agents. For example, in a model that makes use of a centralized repository of reputation information, the TRS object would provide that service. TRS objects are useful even in fully decentralized systems, if only to coordinate trust-related actions. For example, when a buyer requests reviews from other buyers, this request would be processed through the TRS, which co-ordinates such communication.

Some important points should be made regarding TRSes. First, as depicted in the diagram, multiple TRSes may be in use simultaneously, for example, by a heterogeneous population of agents. Second, in order to implement a system for experimentation, matching TRS objects and entities typically must be developed. The role of each component is dependent on the characteristics of the TRS in use. For example, in a completely decentralized model, entities may do all reputation tracking and computation; in this case, the TRS might simply serve as the communication channel between agents. At the other extreme, with a completely centralized model, the TRS may perform all reputation-related functions, while entities simply make use of the services provided. This model seems to provide a great deal of flexibility, without undue complication. Third, in some cases (e.g., a market-wide mechanism), a TRS is tightly integrated into the operation of the marketplace itself. For example, Basic Trunits [5] controls what offers may be made by sellers. TRS objects interface with the Marketplace to allow this.

Note that no particular trust representation, or communication protocol is enforced between agents. In fact, it is up to the designer of the TRS component, along with the associated agents, to determine exactly how (or if) communication takes place between agents. This provides support for a wide range of approaches to trust. Note, too, that communication between heterogeneous agents can be supported. Certainly, mapping from one agent's trust representation to another's may be necessary, but the method for doing so is up to the designer of the TRS component used for communication. (This, in turn, also allows experimentation with different means of allowing heterogeneous agents to interact.)

Another important feature of the testbed is the separation of the agent roles into two components: accounts and entities. *Accounts* represent actual user accounts within the marketplace; these are the identities that are observable by other parties in the market. *Entities*, however, represent the actual agents performing actions by using the accounts. Entities are not observable by marketplace participants, reflecting the fact that identity is difficult to establish, particularly in large electronic marketplaces. This distinction is important for a number of reasons. It allows the re-entry phenomenon to be incorporated into experiments (where agents can simply open new user accounts to shed a disreputable identity). It allows for a single agent to control multiple user accounts (as they may in real-world scenarios), as used in attacks demonstrated in our study [6]. It also allows for investigation of collusion. In the case of perfectly loyal and coordinated collusion, a single entity can represent the entire coalition; in the case of

less perfect coalitions, entities can be implemented that communicate with one another outside the observable marketplace. Note that although buying and selling entities are shown as distinct in our architecture diagram, a single entity can play both roles—for example, when controlling both buying and selling accounts to engage in ballot stuffing.

For different components of the marketplace testbed to communicate with one another, certain aspects of communication must be standardized. In our formulation, communication for actual market transactions (i.e., actual purchases) is defined by the specification: the syntax and semantics of product offers, offer acceptances, etc. These are items that are likely to be standardized in a real marketplace situation. Note, however, that the characteristics of communications between TRS components (e.g., exchange of reputation information between agents) are not imposed by the specification, instead left to be determined by those implementing TRS/agents for the system. This ensures maximum flexibility and fair treatment of different TRS approaches, without undermining the standardization of the TRS proposal. Using this architecture, we evaluated five noteworthy TRSes [3,4,5,7,8]. These models use a variety of approaches, including direct experience, witness information, and centralized mechanisms; for each model, agents were able to represent and communicate trust in the native form as described by the authors, without conforming to a specification imposed by the testbed. This demonstrates the versatility of the platform.

Not shown in the diagram is the StatsKeeper module. StatsKeepers are used to accumulate data from runs of the testbed. A default StatsKeeper implementation was used in our experiments, and would be provided as part of this testbed. This module accumulates sales/profit/cheating statistics, by agent group, over time during testbed execution. (An chart illustrating the data accumulated by this module is provided later in the paper.) New StatsKeeper modules can also be easily developed. All objects in the marketplace are visible to StatsKeepers; full data for every sale executed is provided by the Marketplace to the StatsKeeper, so the desired data can be extracted.

Several architectural details are worth noting:

- Buyers do not know of selling accounts until that seller makes an offer. Sellers do not know of the existence of a buying account until it makes itself known by accepting an offer.
- At the time of making an offer, sellers do not know or control whether an offer will be accepted, or by whom.
- A seller can only provide products that she is able to produce. A seller is able to *advertise and sell* (dishonestly) any product, however.
- It is possible for an agent to connect to multiple TRS objects. This allows experimentation with situations where, for example, an agent might make use of shared reputation information with trusted neighbours, as well as accessing data in a centralized repository (i.e., a different TRS).
- Entities can create new accounts at will.

### 3.4 Simulation Execution

Each round represents one day. Each round consists of the following steps (coordinated by the System Controller):

1. After entering into a sale, a buyer will not know whether or not he has been cheated until after some number of days has passed, reflecting processing, shipping, etc; we refer to the rendering of feedback after this *lag* (14 days) as the *completion* of the sale. At the beginning of each day, buyers are notified whether each completing sale was executed honestly or not.
2. After learning about the outcome of each completing transaction, the buyer determines its satisfaction with the transaction (and submits it to the TRS, if using a system that requires immediate reporting.)
3. Upon receiving feedback (if the TRS requires it), the TRS processes incoming feedback.
4. Each buyer's needs are determined for the day. Each buyer is randomly assigned a set of products (up to 5) that it needs to purchase that day; again, these are selected using the right half of a Gaussian distribution, so there is a greater likelihood of needing lower-priced items.
5. Sellers make offers, submitting them to the marketplace. No limits are placed on sellers' capacity or inventory. If a TRS in use regulates market activity, the marketplace consults the TRS for the validity of each order, before posting it.
6. Buyers select the products they wish to purchase, from which sellers, by consulting the posted offers. Buyers are free to consult their TRSes in so doing. For each purchase they decide to make, an offer acceptance is communicated to the corresponding seller account, via the marketplace.
7. Sellers receive the offer acceptances, and have the opportunity to decide if they wish to complete each such sale. Sellers may consult their TRSes to make the decision. For each sale that the seller agrees to make, it decides whether or not to fulfill it honestly or dishonestly. Acceptances are communicated to the marketplace, which forwards each to the corresponding buyer account.
8. Payment is transferred from the buyer account to the seller account, for each sale.
9. Each sale's status (honest or dishonest) is communicated by the seller to the marketplace for storage. This value is not observable by any other marketplace participant, until the buyer is notified during Step 1 of a later round.

### 3.5 Initial Test Set

Much of the value to be gained from an evaluation testbed such as this is the value of the tests: their difficulty, their breadth, and their representativeness of the sorts of issues TRSes might face in a real environment. As an initial set of tests, we provide the set of agents used in our earlier investigations [6]. These agents employ a number of different tactics (consisting only of honest/dishonest transactions that can be executed within the marketplace) based in part on the

problems described in [5]. These agents were designed to test the robustness of TRSes that attempt to cope with dishonest sellers; as such, each agent described below is a seller. They seek to cheat profitably, despite the use of the TRS in question. This test set is far more extensive and difficult than any we have seen used for evaluation of TRSes to date.

The test set includes the standard, randomly cheating agents employed in so many evaluations. Beyond this, it includes the following agents:

- The **Proliferation agent**, who seeks to win an abnormally large portion of sales by offering products through many user accounts simultaneously, crowding competitors out of the market.
- The **Reputation Lag** agent depends on the delay that exists in many marketplaces between the time he is paid by the buyer, and the time that his reputation is updated to reflect the outcome of the transaction. This agent seeks to build a positive reputation, then use this reputation to cheat as many buyers as possible in the brief period before his reputation is updated to warn other buyers of the change in behaviour.
- The **Re-entry agent**, who creates a new account, attempts to use the non-disreputable status of this new account to cheat as many buyers as possible, then abandons the account and creates a new one, to begin the cycle anew.
- The **Value Imbalance agent**, who seeks to gain good reputation through honest small-value sales, then use that reputation to cheat buyers on large-value sales.
- The **Multi-tactic agent**, who knows how to use all of the tactics described above, and attempts to profitably wield the entire portfolio. This agent is especially intended to undermine the notion of 'security by obscurity', that a TRS might be safe from such tactics if the would-be cheater doesn't know which TRS is in use (and hence its specific vulnerabilities).

As demonstrated in [6], this set of attacks was quite devastating to the set of TRSes evaluated—all of the TRSes were defeated by numerous attacks, and none withstood the Multi-tactic agent.

In actual implementation, we found it useful to decompose our entities into two parts: the actual entity itself, and tactic modules. Each tactic module contains a particular behaviour, for example, the method for launching a particular attack. An entity, then, makes use of one or more tactics to execute its trading activity. This design allows tactics to be re-used. More importantly, it facilitates the design of agents that employ multiple tactics. As noted above, our multi-tactic agents provide an extremely difficult (but realistic and practical) test for TRSes.

Figure 2 depicts one run of the testbed, pitting the Basic Trunits TRS against the Multi-tactic agents. This figure illustrates some of the data generated by the default StatsKeeper module. In this chart, 'smart' agents are those employing the multi-tactic approach. Honest sellers, and sellers who cheat randomly are included for comparison. The dashed lines represent the revenue from sales for the day in question (smoothed for presentation), while the solid lines represent profit (i.e., revenue less the cost incurred to furnish the goods). There are an

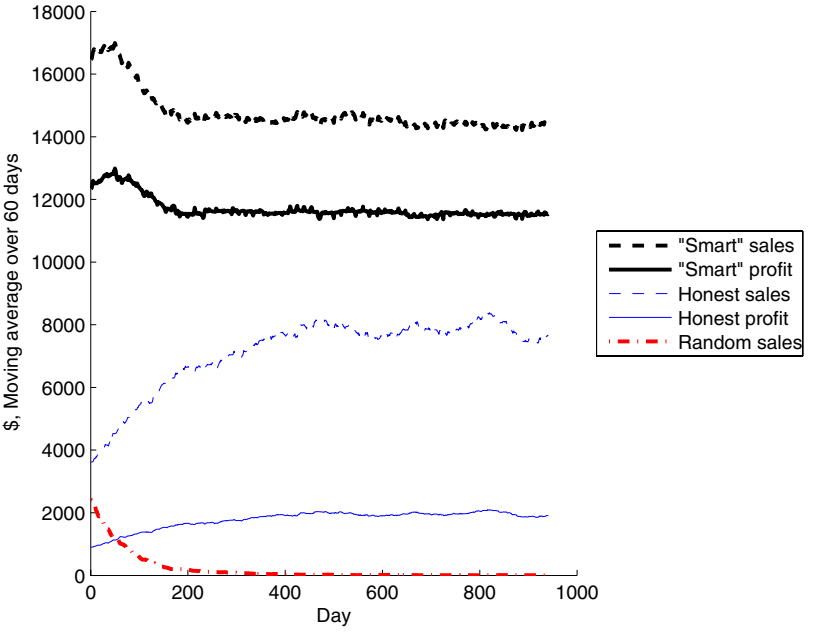


**Fig. 2.** Basic Trunits, vs. Multi-tactic cheating sellers

**Table 1.** Sales/profit for sellers using multiple attacks

| **TRS** | **Cheater sales (% of honest)** | **Cheater profit (% of honest)** |
|---|---|---|
| Tran & Cohen | 1775.3% | 6765.1% |
| Beta | 107.1% | 288.3% |
| TRAVOS | 274.6% | 613.0% |
| Yu & Singh | 274.9% | 723.4% |
| Basic Trunits | 181.8% | 577.7% |

equal number of agents in each group. Note that the multi-tactic agents are far more profitable than the other groups—cheating is by far the most profitable policy, meaning that the TRS has failed this test.

Table 1 shows the results obtained by the multi-tactic agent against all of the TRSes tested. The first column in each table represents the average sales (in dollars) per multi-tactic cheating agent, relative to those of an honest seller. The second column reflects the profit realized by a cheating agent, relative to an honest one. (Sales are generally more profitable for cheating agents, because they do not incur the cost of honestly furnishing the good.) Results greater than 100% mean that the average multi-tactic agent makes more money than an honest agent. For example, a value of 124% would mean that cheating agents earned 24% more than honest agents per capita. Here, the cheating agents make

far more money than honest ones—quite troubling, because this means that a profit-maximizing agent should choose to cheat rather than be honest. All of the TRSes have failed the test.

This set of tests is certainly not exhaustive. Further, it only tests TRSes against dishonest sellers, rather than dishonest buyers, agents that lie to one another about their opinions of other agents, etc. Nonetheless, it constitutes a substantial initial test suite for this important aspect of TRS operation.

## 4 Conclusions and Future Work

The trust and reputation testbed described in this paper allows a breadth of experimentation and a thoroughness and objectivity of evaluation that have previously been unavailable from publicly-available, standardized testing tools. The design of this testbed is a proven one, having been used to shed light on important issues that had previously been unexplored experimentally—in particular, the degree to which existing TRS proposals can withstand cheating agents that actively attempt to circumvent the protections of the system. The platform has shown itself to be flexible, supporting experimentation with TRSes using a variety of approaches.

With this design, we have attempted to allow the greatest degree of flexibility of experimentation possible, while still providing ease-of-use and the ability to generate meaningful benchmarks and comparisons. As a result, not every TRS can be tested using this platform: for example, ones that do not function in marketplace environments.

The testbed is intended to allow more thorough testing than has typically been performed for TRSes in the past. As TRS researchers develop new agents, the test suite will grow, increasing the rigour of the evaluations, and the insights provided.

While we believe this testbed to be an important tool in itself, and a significant step towards improving the evaluation of TRSes, we do not contend that it is perfect or complete in its current formulation. We hope that this proposal serves as a basis for discussion and development within the trust and reputation community. The value of this tool will increase with input from other researchers, ensuring that the platform is complete, and flexible enough (within the limits of practicality) to meet their needs. Future work includes consultation with researchers to identify potential refinements and achieve consensus regarding:

- Scenario;
- Architecture;
- Recommended parameter values for benchmarking;
- Range of support for agents, and trust and reputation technologies.

Through these consultations, we will be diligent to ensure an effective balance between the complexity that might be introduced with increased flexibility, and the standardization that facilitates benchmarking and usability.

## References

1. 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: AAMAS 2005: Proceedings of the fourth international joint conference on Autonomous Agents and Multiagent Systems, pp. 512–518. ACM, New York (2005)
2. Teacy, W.T.L., Huynh, T.D., Dash, R.K., Jennings, N.R., Luck, M., Patel, J.: The ART of IAM: The winning strategy for the 2006 competition. In: Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007) Workshop on Trust in Agent Societies, Honolulu, Hawaii, USA (2007)
3. Teacy, W.T., Patel, J., Jennings, N.R., Luck, M.: Travos: Trust and reputation in the context of inaccurate information sources. Autonomous Agents and Multi-Agent Systems 12(2), 183–198 (2006)
4. Yu, B., Singh, M.P.: Distributed reputation management for electronic commerce. Computational Intelligence 18(4), 535–549 (2002)
5. Kerr, R., Cohen, R.: Modeling trust using transactional, numerical units. In: PST 2006: Proceedings of the Conference on Privacy, Security and Trust, Markham, Canada (2006)
6. Kerr, R., Cohen, R.: Smart cheaters do prosper: Defeating trust and reputation systems. In: Proceedings of the Eighth International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009), Budapest, Hungary (2009)
7. Jøsang, A., Ismail, R.: The beta reputation system. In: Proceedings of the 15th Bled Electronic Commerce Conference e-Reality: Constructing the e-Economy (June 2002)
8. Tran, T., Cohen, R.: Improving user satisfaction in agent-based electronic marketplaces by reputation modelling and adjustable product quality. In: AAMAS 2004: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems, Washington, DC, USA, pp. 828–835. IEEE Computer Society, Los Alamitos (2005)
9. Hang, C.W., Wang, Y., Singh, M.P.: An adaptive probabilistic trust model and its evaluation. In: AAMAS 2008: Proceedings of the 7th international joint conference on Autonomous Agents and Multiagent Systems, Richland, SC, International Foundation for Autonomous Agents and Multiagent Systems, pp. 1485–1488 (2008)
10. Harbers, M., Verbrugge, R., Sierra, C., Debenham, J.: The examination of an information-based approach to trust. In: Noriega, P., Padget, J. (eds.) International Workshop on Coordination, Organization, Institutions and Norms (COIN), pp. 101–112. Durham University, Durham (2008)