A Framework for Agent-Based Trust Management in Online Auctions[*]

Haiping Xu[†], Sol M. Shatz[‡], and Christopher K. Bates[†]
[†]*Computer and Information Science Department*
University of Massachusetts Dartmouth, North Dartmouth, MA 02747, USA
E-mail: {hxu, u_cbates}@umassd.edu
[‡]*Computer Science Department*
University of Illinois at Chicago, Chicago, IL 60607, USA
E-mail: shatz@uic.edu

Abstract

Current electronic commerce applications such as online auction systems are not trustworthy due to a lack of effective trust management mechanisms. A trustworthy online auction system requires a dynamic trust management module that can detect abnormal bidding activities in real-time, notify the involved users, and cancel the corresponding auction immediately. In this paper, we present a general framework for agent-based trust management (ATM) in online auctions. The ATM module consists of three types of agents, namely the monitoring agent, the analysis agent and the security agent. A monitoring agent can monitor a bidder and detect any abnormal bidding behavior; while the analysis agent and the security agent can analyze state-based information and history information of a bidder, and make decisions on shill detection, respectively. We illustrate the communication protocol among various agents, and demonstrate our agent-based trust management approach for online auctions using a prototype ATM module developed with JADE.

1. Introduction

There have been many implementations of online auction houses, such as Firstauction (Firstauction.com), uBid (uBid.com), and eBay (eBay.com); however, current online auction systems are still suffering from a major weakness: *lack of trustworthiness*. In a recent research study, computer scientists at Carnegie Mellon University used data mining techniques to analyze historical auction data from eBay, and detected more than a dozen probable fraudsters and several dozen apparent accomplices [1]. Most of the fraud detected in online auctions is related to shilling behaviors, which occurs when a seller disguises himself as a legitimate bidder by using a second identity or account solely for the purpose of pushing up the sale price [2, 3].

To develop a trustworthy online auction system, it is vital to introduce feasible trust management mechanisms to prevent, detect and avoid trading fraud such as shilling behaviors. Trust management can be defined as the activity of collecting and analyzing security relevant evidence, and making assessments and decisions on trust relationships for e-commerce transactions [4]. During the past decade, trust has been studied in terms of decentralized access control, public key certification, and reputation systems for Peer-to-Peer (P2P) systems. There exist two major approaches for trust management, namely policy-based and reputation-based trust management [5]. These trust management approaches treat trust as independent of context and time, thus they do not support real-time monitoring and dynamic trust management. Auction houses using these models may help to build the trustworthiness of an auction house by detecting undesired activities based on historical auction data; however, it is very limited in helping to avoid users' losses – in terms of money and time – due to shilling behaviors. A trustworthy online auction system requires a dynamic trust management system that can detect abnormal bidding activities in real-time, notify the involved users, and cancel the corresponding auction immediately. Such treatment requires a strong and secure model to be established in order to provide a secure environment for online transactions, through which users can safely complete their transactions, build and maintain trust relationships among the users as well as with the auction house.

In order to develop an effective dynamic trust management model for online auction systems, the behaviors and the current state of the trustee (i.e., the users) must be considered. In our approach, we use an agent-based trust management (ATM) module to monitor, analyze and detect shill bidders dynamically. The ATM module consists of three types of agents, namely the monitoring agent, the analysis agent and the security agent. A security agent can dispatch a number of monitoring agents to detect suspicious users; meanwhile, an analysis agent can analyze users' bidding behaviors using real-time auction data and historical information. Based on the analytical results, the security agent can decide on whether a suspicious shill is an actual shill, and may send a request to an auction agent to cancel the involved auction.

2. Related work

Trust management using reputation models are based on prior history of users and/or feedback gathered from other entities. Shmatikov and Talcott proposed a formal model that precisely defined the notion of reputation, and can be used to reason about trust [6]. Based on the license-based digital rights language, they used licenses to formalize both "good" and "bad" behaviors, which specify obligations and forbidden actions, respectively. Selcuk and his colleagues proposed a reputation based trust management protocol for P2P networks, where users rate the reliability of other parties, and share this information with their peers [7]. Recently, reputation based trust management has been applied to sensor networks. For example, Ganeriwal and Srivastava proposed a reputation-based framework for sensor networks (RFSN), which allows development of a community of trustworthy sensor nodes based on their behaviors, and can maintain the reputation for sensor nodes and evaluate their trustworthiness [8].

Trust and reputation management has been a promising approach to building trustworthiness in networked systems. However, many reputation models suffered from a major drawback – there is no effective mechanism to prevent users from giving false information when making a recommendation. In addition, this approach fails to evaluate the trustworthiness of new users or those who have not yet received enough reputation feedback. As a result, reputation based trust management models are usually *not* sufficient to ensure the trustworthiness of a system. On the other hand, policy-based trust relies on trusted certification authorities (CA) and signed certificates as well as access control policies to determine whether a requester is trusted or should not be allowed to access a certain resource [5]. Declarative policy rules are well suitable for specifying access control conditions for access permissions to certain resources. Policy languages such as REFEREE [9] and KeyNote [10] can support authorizing the trustee automatically by determining whether certain credentials are sufficient for performing a certain action or accessing a certain resource. At the authentication level, trust is bound to a certain identity or membership, which is not changeable during a period of time. The focus of such an approach is on credential matching policy. However, no research has yet emerged for updating the level of trust based on evidence of actual behaviors in real-time [11]. In some recent work, Bonatti and his colleagues proposed an integrated trust management framework that combines rule-based and credential-based trust, which is capable of addressing the complexity and the variety of semantic web scenarios with both structured organizational environments and unstructured user communities [5]. Most of the existing work on trust management emphasized defining static mechanisms or algorithms to calculate users' trustworthiness value. The evolution of trust has been seldom discussed in detail in practical contexts due to the difficulty in reconfiguring the trust management model based on newly acquired evidence [11]. In contrast, we propose an agent-based trust management (ATM) module that facilitates real-time trust re-evaluation by updating user roles and access permissions dynamically. As a result, our approach provides a better foundation towards building a trust-worthy networked system. In this paper, we introduce our ATM module in the context of online auction systems, which require strong trustworthiness of the auction house with true and timely evaluation of users' bidding activities. The approach presented in this paper is based on our previous work for real-time trust management in agent-based online auction systems [12]; however, our previous trust management model only consists of a security agent that is responsible for all tasks of shill detection, which is *not* scalable when the number of users increases dramatically. To solve this problem, in this paper, we define the ATM module as a multi-agent system. In order to efficiently detect shill bidders, monitoring agents can run concurrently and use real-time auction data to search for shill suspects based on patterns of shilling behaviors. When a shill suspect is detected, the security agent and the analysis agent can use available user information and auction data to verify whether the shill suspect is an actual shill. Furthermore, unlike our previous work, the ATM module introduced in this paper is a generic module that can be applied in both agent-based online auction systems and conventional online auction houses such as eBay and Yahoo!Auctions.

3. Agent-based trust management

3.1. Trustworthy online auction house

A multi-agent system (MAS) consists of a number of software agents that can work autonomously, but need to coordinate with each other to accomplish tasks and missions. Each agent is built with enough capability to work independently. The coordination model based on asynchronous message passing among agents provides a uniform interface for their interaction; while the mechanism of storing and routing messages enhances the fault tolerance capability of the whole system. Figure 1 is an overview of an agent-based trustworthy online auction house. The auction house is a multi-agent system, which consists of a main agent, an agent-based trust management (ATM) module, and a number of auction agents. The main agent manages the auction house and provides an interface for the auction house administrator to manipulate and monitor the auction house. The main agent is responsible for initializing the ATM module and generating an auction agent for each auction started. A user or a bidder can join one or more than one auction at the same time, and put in bids on auctioned items concurrently. All auction data is stored in a local or remote auction database.

The agent-based trust management (ATM) module in an auction house is the key component for maintaining trustworthiness of the online auction system. The major tasks of the ATM are to detect shills and update the trust levels of the shill bidders. When a shill bidder is detected, the ATM informs the responsible auction agents, and the auction agents will then notify all involved users and cancel the corresponding auctions immediately.
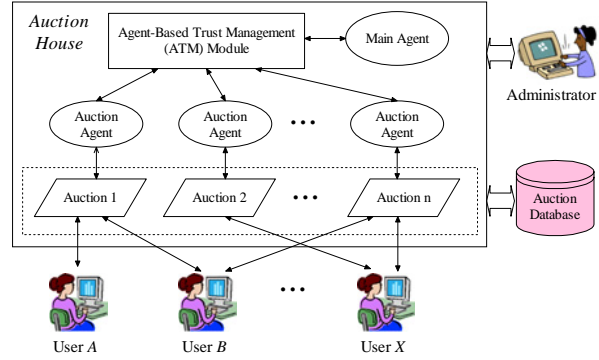

Figure 1. A trustworthy online auction house

Note that the trust level of a user is defined based on a user's role level as described in Section 3.2. In addition to downgrading the trust level of a user when such a user is detected as a shill, the ATM can also upgrade the trust level of a user when there is sufficient evidence showing that the user is a trusted one.

3.2. Agent-based trust management module

Figure 2 shows the general architecture of the agent-based trust management (ATM) module for online auction systems, which consists of three types of agents, i.e., the security agent, the analysis agent and the monitoring agent. From Figure 2, we can see that before a user can start trading in an auction, she must first be authenticated to get the initial pass. The authentication process is based on the user's credential as well as her user name and password. After the authentication process is completed, the authorization process starts. In our ATM module, we adopt the role-based access control (RBAC) mechanism to effectively
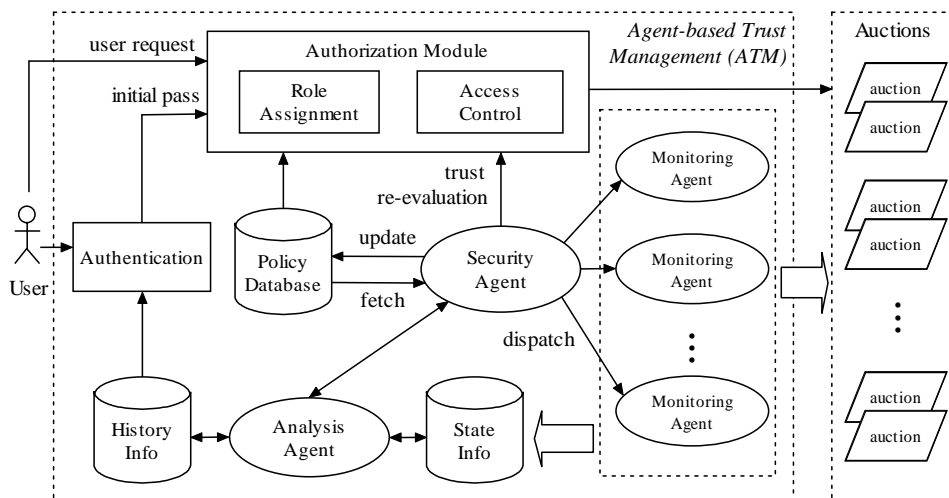

Figure 2. Agent-based trust management (ATM) module

deploy user authorization and access rights. The RBAC model has been proposed as one of the most effective solutions to providing security features in different distributed computing infrastructure [13]. In an RBAC model, users are assigned roles with permissions, which are access modes that can be exercised on a particular object or service in the system. RBAC ensures that only authorized users are given access to certain data or resources. Most of the RBAC models follow the same basic structure of subject, role and privilege. However, in a more sophisticated role-based access control model, access decisions for an application depend on the combination of a user's credential, the context, the system state, and other factors such as relationship, time and location. The RBAC mechanism adopted in our ATM module not only depends on a user's identity and credential, but also depends on the user's current state and the user's history information. In the ATM module, we can define certain policy rules that specify and update a user's access right to online auctions with certain permissions (restrictions) based on the user's previous and current states. Thus, our role-based access control approach defines a *stateful* mechanism.

Similar to the trust management module introduced in our previous work [12], the *Authorization Module* in the ATM consists of two major components, namely the *Role Assignment* module and the *Access Control* module. The *Role Assignment* module is used to assign a user a role to participate in online auctions. For example, a "Seller" role allows a user to sell auction items, while a "Bidder" role allows a user to submit bids on an auctioned item. The trust levels of a user are defined by the sub-role classes that the user belongs to. For example, we categorize a "Bidder" role into five different types of sub-roles, i.e., "MostTrustedBidder," "TrustedBidder," "NeutralBidder," "UntrustedBidder," and "MostUntrustedBidder". A "Seller" role is also categorized into 5 sub-roles in a similar way. The role assignment is supported by the *Policy Database*, which contains two types of policies: policies for role assignment, called *RA-Policy*, and policies for role-based access control, called *AC-Policy*. Table 1 lists two examples of *RA-Policy*, which are written in the policy language PROTUNE [5].

In Table 1, the RA-Policy *A* says that if a requester is a new user, and the request type is "buy", then the requester will be assigned a role of "NeutralBidder." Most of the known auction systems today use static role assignment for users. However, static role assignment is not suitable for real-time trust management since trust is dynamic by nature and may change at runtime. In our approach, a user's role is dynamic and can be re-assigned at runtime based on the security agent's view as well as newly captured

evidence. In Table 1, the RA-Policy *B* shows that a requester's role can be changed from "NeutralBidder" to "UnTrustedBidder," if the requester's current shilling score is no less than 0.6, and during a month, the requester's reputation score is no more than 0.7. From the above examples, we can see that the *Role Assignment* module assigns a role to a requester based on various factors, such as the type of request from the user, current role assignment, current shilling score and reputation score, as well as historical information about the user.

Table 1. Examples of RA-Policy

RA-Policy: A

assignRole(Requester, NeutralBidder) ←
   newUser(Rquester),
   requestType(Requester, buy).

RA-Policy: B

changeRole(Requester, UnTrustedBidder) ←
   currentRole(Requester, NeutralBidder),
   shillingScore(Requester, X, current), $X \geq 0.6$,
   reputationScore(Requester, Y, oneMonth), $Y \leq 0.7$.

Table 2. Examples of AC-Policy

AC-Policy: A

allow(Requester, Bid) ←
   ¬newUser(Rquester),
   currentRole(Requester, TrustedBidder),
   shillingScore(Requester, X, current), $X \leq 0.3$,
   reputationScore(Requester, Y, current), $Y \geq 0.6$.

AC-Policy: B

disallow(Requester, Bid, oneWeek) ←
   ¬newUser(Rquester),
   currentRole(Requester, UnTrustedBidder),
   shillingScore(Requester, X, current), $X \geq 0.6$,
   reputationScore(Requester, Y, current), $Y \leq 0.7$.

Once the role assignment (or role re-assignment) process is done, the *Access Control* module will be invoked. Similarly, the *Access Control* mechanism is also supported by the *Policy Database*, which stores a set of *AC-Policy*. Table 2 lists two examples of such policies. AC-Policy *A* says that a requester is allowed to bid if the requester is not a new user, its current role is "TrustedBidder," its current shilling score is no more than 0.3, and its current reputation score is no less than 0.6. Note that if a requester is a new user, she is always allowed to bid as long as she has passed the authentication process. Similarly, AC-Policy *B* says that if a requester with an "UnTrustedBidder" role has a current shilling score of no less than 0.6, and a current reputation score of no more than 0.7, then the requester shall be disallowed to bid in any auctions for a week. In the above examples, the threshold values for the variables are initially subjectively determined by the auction house administrator, but they can be improved and updated at runtime by the security agent that has a learning capability.

The security agent is responsible for updating the policy database, providing trust related information, and reasoning about role-based authorization. One of the challenging tasks in developing the security agent is to define a security agent inference model, which deals with incomplete information and uncertainty in online auctions. To derive reasonable trust values of a user, the security agent may need to "make a guess" about a user's objectives and the user's real intention. For example, a user who is detected as a suspicious shill may have no intention at all to be a shill. In this case, the security agent should conclude the user as a normal bidder. Thus, how to define an effective user's intention model becomes a very important issue.

Reasoning under uncertainty has been one of the major topics of our ongoing research for this project. Further details on this topic are outside the scope of focus for this paper. Other information, such as a user's shilling score and reputation score, are calculated by the analysis agent. Although it is not straightforward to design efficient algorithms that support analysis of real-time auction data, we have demonstrate some preliminary results of calculating shilling score using modeling checking techniques [14]. The reputation score of a user can be calculated as an accumulated value of the ratings or feedback from other users. The major responsibility of a monitoring agent is to detect shill suspects by matching bidding activities with shilling patterns. The monitoring agent can inform the security agent about the shill suspect in order to verify if it is an actual shill. To detect shill suspects, the monitoring agent needs to access real-time auction data that is captured by auction agents. When an auction takes place on a different machine from the one where the monitoring agent resides, the monitoring agent can migrate from its local host to a remote host. When the monitoring agent completes its task, it sends back the needed information to the *State Info* database (as shown in Figure 2). In addition to the real-time auction data, the *State Info* database also stores the latest state information about a user, e.g., a user's current role assignment, current access permissions, current shilling score and reputation score. On the other hand, the *History Info* database (as shown in Figure 2) stores prior information (e.g., a user's previous reputation scores) about a user. When the security agent makes a decision, for example to change a user's role assignment, the security agent will notify the analysis agent to update the *History Info* database. The *History Info* database also serves as a knowledge base for analysis purposes. For example, when the security agent learns a new bidding pattern, it will pass such information to the analysis agent. The analysis agent will then check the consistency of the newly derived pattern with existing bidding patterns stored in the database. If there are no conflicts, the new bidding pattern will be added into the *History Info* database for future reference.

3.3. Agent communication in ATM module

Software agents typically use asynchronous message passing for agent communication. One of the major agent communication standards is called FIPA-ACL (Foundation for Intelligent, Physical Agents – Agent Communication Language). FIPA-ACL is grounded in speech act theory, which states that messages represent actions or communicative acts – also known as speech acts or performatives [15]. FIPA-ACL defines a set of 22 communicative acts, such as inform, request, agree, not understood, and refuse. In Figure 3, we use the UML sequence diagram to illustrate the communication protocol for shill detection among various agents, namely the security agent (SecurAgent), the analysis agent (AnalyAgent), the monitoring agent (MonAgent), and the auction agent (AucAgent).
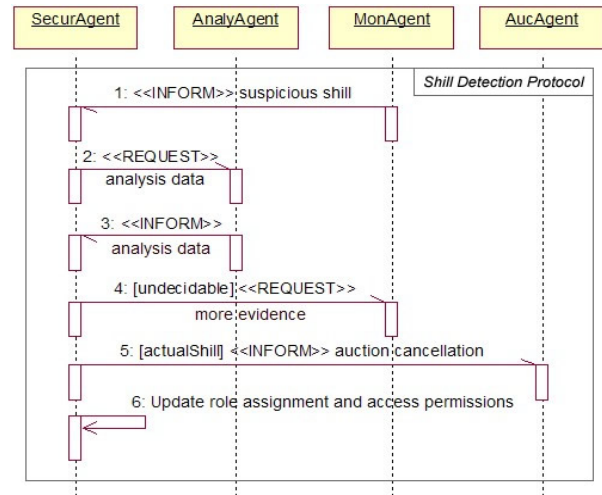

Figure 3. Interaction protocol for shill detection

From the diagram, we can see that the monitoring agent first informs the security agent about a suspicious shill, say bidding agent *B*1. Upon receiving this message, the security agent requests analytical data, such as shilling score and reputation score of B1, from the analysis agent. After analyzing the auction data as well as the history information of *B*1, the analysis agent informs the security agent about the analytical results. When the security agent receives the analytical data, it verifies if the suspicious shill is an actual shill. In case the security agent will not be able to make such a decision, it will request more evidence from the monitoring agent. In this case, steps 1-3 of the

communication protocol must be repeated (for simplicity, we did not show the loop in Figure 3). When an actual shill is detected, the security agent informs the involved auction agent to cancel the affected auction, and it also updates the role assignment and access permissions of the shill bidder.

Note that although we show only one monitoring agent and one auction agent in Figure 3, the security agent can communicate with multiple monitoring agents and auctions agents concurrently. Furthermore, it is also possible that a shill bidder is detected for his concurrent shilling behaviors in multiple auctions. In this case, more than one auction needs to be canceled.

4. Prototyping the ATM module

To illustrate that various agents from the agent-based trust management module can effectively communicate with each other using FIPA-ACL based interaction protocols, we developed a prototype ATM module using JADE (Java Agent DEvelopment framework) [15]. The JADE framework facilitates the development of complete agent-based applications by means of a run-time environment implementing the life-cycle support features required by software agents. JADE is fully compliant with the FIPA-ACL specification. An example of an FIPA-ACL message can be illustrated as follows.

```
(INFORM
 :sender (agent-identifier :name MonAgent-2
 @PT502989:1099/JADE :addresses
 (sequence http://192.168.1.100:7778/acc))
 :receiver (set (agent-identifier :name
 SecurAgent@PT502989:1099/JADE :addresses
 (sequence http://192.168.1.100:7778/acc)))
 :content "Suspicious shill B2 detected!"
 :language "Plain English"
 :ontology "Online Auctions"
 :protocol "shill Detection Protocol"
 :conversation-id inform-shill-suspects)
```

The above message is sent by monitoring agent *MonAgent-2*, which informs the security agent *SecurAgent* that the bidder *B*2 is a suspicious shill. Upon receiving this message, the security agent will be responsible for verifying if the shill suspect *B*2 is an actual shill. Figure 4 shows the interface of the security agent *SecurAgent* for ATM. In our example, we have set up two concurrent auctions *A*1 and *A*2, which are managed by two auction agents *AucAgent-1* and *AucAgent-2*, respectively. We have also arranged three bidders, *B*1, *B*2 and *B*3, whose current roles are "TrustedBidder," "NeutralBidder," and "MostTrusted-Bidder," respectively.

From Figure 4, we can see that the security agent first dispatches three monitoring agents, *MonAgent-1*,

MonAgent-2, and *MonAgent-3*, to monitor the three bidders *B*1, *B*2, and *B*3 by watching their bidding activities at runtime. When the bidding activities of a bidder match with a predefined shilling pattern, the corresponding monitoring agent will immediately inform the security agent that the bidder is a suspicious shill. In our demonstration example, the monitoring agent *MonAgent-2* detects that *B*2 is a suspicious shill, and it sends an INFORM message to the security agent immediately for further analyses. When the security agent receives this message, it sends a REQUEST message to the analysis agent *AnalyAgent* for the analytical data associated with bidder *B*2. By analyzing *B*2's state information and history information, the analysis agent calculates the reputation score and shilling score for bidder *B*2, and sends the analytical results as an INFORM message to the security agent. Based on *B*2's current role assignment and the reputation and shilling score, the security agent uses its reasoning mechanism and concludes that the bidder *B*2 is an actual shill. Since bidder *B*2 showed his shilling behavior in auction *A*1, auction *A*1 must be cancelled in order to protect the interests of other bidders who are also involved in auction *A*1. The security agent notifies the auction agent *AucAgent-1* about the auction cancellation, and the auction agent confirms that it received this notice and will notify all involved bidders about the auction cancellation.
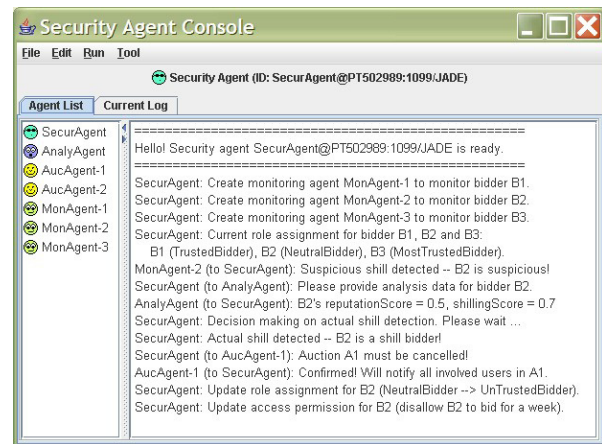

Figure 4. Security agent interface for ATM

Figure 5 shows the actual messages passed among various agents in ATM module. The messages are captured by a special agent called *sniffer* agent, provided by the JADE framework for debugging and documenting conversations between agents. Our demonstration example shows that agents defined in our ATM module can properly communicate with each other using standard FIPA-ACL communicative acts.
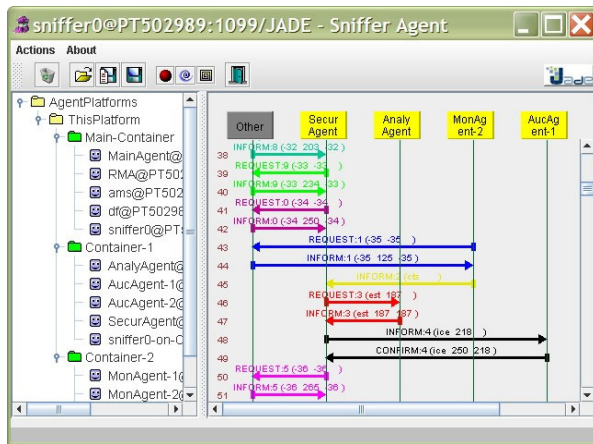
Figure 5. Messages captured by the sniffer agent

5. Conclusions and future work

Electronic commerce plays an important role in the worldwide economy. However, current forms of electronic commerce are unsafe due to a lack of effective trust management mechanisms. This weakness is quite severe in online auction systems because shill bidders can easily disguises themselves as legitimate users in order to drive up the sale prices for auctioned items. In this paper, we introduced a framework for agent-based trust management (ATM) module that supports trustworthy computing in online auctions. Our ATM approach facilitates real-time detection of suspicious shills by monitoring agents, analysis of auction data by the analysis agent, and verification of actual shills by the security agent. The approach also supports dynamic updating of role-assignment and access permissions for bidders. Thus, our approach provides a strong and secure model for development of trustworthy online auction systems. Our prototype ATM developed using JADE illustrates that the agents in the ATM module can effectively communicate with each other in order to detect actual shills and update users' trust level denoted by different types of roles and various access permissions. For our future work, we will study and accommodate approaches for verification of actual shills with incomplete knowledge and uncertainty into our ATM framework. We plan to build a bidder's intention model and make decisions on detection of actual shills with a Bayesian network learned from historical auction data and information about users.

6. References

[1] CMU, "Online Auction Fraud: Data Mining Software Fingers Both Perpetrators and Accomplices," *ScienceDaily*, December 5, 2006.

[2] W. Wang, H. Zoltán, and A.B. Whinston, "Shill Bidding in Multi-Round Online Auctions," In *Proceedings of the 35th Hawaii International Conf. on System Sciences* (HICSS'02), Hawaii, 2002.

[3] J. Trevathan and W. Read, "Detecting Collusive Shill Bidding," In *Proceedings of the Fourth International Conference on Information Technology: New Generations* (ITNG 2007), Las Vegas, Nevada, USA, April 2007, pp. 799-808.

[4] T. Grandison and M. Sloman, "Trust Management Tools for Internet Applications," In *Proceedings of the 1st International Conference on Trust Management*, May 2003, Crete, Springer LNCS 2692, pp. 91-107.

[5] P.A. Bonatti, C. Duma, D. Olmedilla, *et. al.*, "An Integration of Reputation-Based and Policy-Based Trust Management," In *Proceedings of the Semantic Web Policy Workshop*, Galway, Ireland, November 2005.

[6] V. Shmatikov and C. Talcott, "Reputation-Based Trust Management," *Journal of Computer Security*, Special Issue on Selected Papers of WITS 2003 (ed. Roberto Gorrieri), Vol. 13, No. 1, 2005, pp. 167-190.

[7] A.A. Selcuk, E. Uzun, and M.R. Pariente, "A Reputation-Based Trust Management System for P2P Networks," In *Proceedings of the 4th IEEE/ACM International Symposium on Cluster Computing and the Grid* (CCGrid 2004), Chicago, USA, April 2004.

[8] S. Ganeriwal and M.B. Srivastava, "Reputation-Based Framework for High Integrity Sensor Networks," In *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks* (SASN 2004), Washington DC, USA, pp. 66-77.

[9] Y. Chu, J. Feigenbaum, B. LaMacchia, P. Resnick, and M. Strauss, "REFEREE: Trust Management for Web Applications," *Computer Networks and ISDN Systems*, Vol. 29, 1997, pp. 953-964.

[10] M. Blaze, J. Feigenbaum, A.D. Keromytis, "KeyNote: Trust Management for Public-Key Infrastructures," In *Proceedings of the Security Protocols: 6th International Workshop*, Cambridge, UK, April 1998, LNCS 1550, Springer-Verlag, 1998, pp. 59-63.

[11] S. Ruohomaa and L. Kutvonen, "Trust Management Survey," In *Proceedings of the iTrust 3rd International Conference on Trust Management*, May 23-26, 2005, Rocquencourt, France, Springer-Verlag, LNCS 3477, May 2005, pp.77-92.

[12] R. Patel, H. Xu, and A. Goel, "Real-Time Trust Management in Agent Based Online Auction Systems," In *Proceedings of the 19th International Conference on Software Engineering and Knowledge Engineering* (SEKE'07), Boston, USA, July 2007, pp. 244-250.

[13] H. Feinstein, R. Sandhu, E. Coyne, and C. Youman, "Role-Based Access Control Models," *IEEE Computer*, Vol. 29, No. 2, 1996, pp. 38-47.

[14] H. Xu and Y-T. Cheng, "Model Checking Bidding Behaviors in Internet Concurrent Auctions," *International Journal of Computer Systems Science & Engineering* (IJCSSE), July 2007, Vol. 22, No. 4, pp. 179-191.

[15] F. Bellifemine, G. Claire, and D. Greenwood, *Developing Multi-Agent Systems with JADE*, John Wiley & Sons. Ltd., 2007.