Design and Simulation of Trust Management Techniques for a Coalition Data Sharing Environment

Srinivasan Iyer
The University of Texas at Dallas
Sxv051100@utdallas.edu

Dr.Bhavani Thuraisingham
The University of Texas at Dallas
bhavani.thuraisingham@utdallas.edu

Abstract

Effective communication among agents in large teams is crucial because the members share a common goal but only have partial views of the environment. Information sharing is difficult in a large team because, a team member may have a piece of valuable information but not know who needs the information, since it is infeasible to know what each other agent is doing. Information sharing is a main part of any system or organization. The information sharing needs to be foolproof. Only the legitimate receiver should be able to get hold of the information. This paper mainly deals with intelligent software agents for information sharing with confidentiality and trust. It clearly defines an Intelligent Software Agent, background of Information sharing in intelligent agents and the trust in the agents. Some part of the information needs confidentiality. The information that is shared requires security policy enforced based on the domain of the information and trust level of individual agent. This paper also provides the results of a multi-agent simulation for sharing information. It also implements trust calculation based on the quality of information provided by the peer agents in the simulation.

1. Introduction

Information sharing is necessary and unavoidable, even in the times of Kings and Empires. There were many alliances between the kingdoms, espionages, miscommunications, treachery, deception, compromises, victories and defeats. The information sharing needs to be secure. That is, it is critical that the information does not get into the wrong hands. Only the legitimate receiver should be able to get hold of the information. Even Kings had their own way of secured Information sharing. They had the royal seals to verify if the information is authentic.

Exciting emerging applications require hundreds or thousands of agents and robots to coordinate to achieve their joint goals. In domains such as military operations, space or disaster response, coordination among large numbers of agents promises to revolutionize the effectiveness of our ability to achieve complex goals. Such domains are characterized by widely distributed entities with limited communication channels among them and no agent having a complete view of the environment. Information relevant to team goals will become available to team members in a spontaneous, unpredictable and, most importantly, distributed way. The question addressed in this paper is when a team member senses some information, how it can decide which team member to communicate that information to. In most applications for very large teams, broadcasting information is not suitable, desirable or feasible. Instead, the agent must attempt to target its information delivery to just the agents that need it. In a large team, each member has a limited model of what other members of the group know or even what many of them are doing. For example, a field agents involved in a military operation may observe many features of a battlefield on route to an assignment. Many of its observations will be relevant to the plans of other combatants but the field agents will not necessarily know which group members require the information.

Since 9/11, the agencies have moved to a need to know paradigm to a need to share paradigm. For many applications it is important that the information be shared and then examine the consequences. There are now efforts on information sharing based on Trust. That is, do I trust say John enough to share some critical information. What happens if I trust John only 50% of the time? Do I still share the information with him? Another good example is coalition data sharing between countries. To fight the global war on terror, organizations have to share data between trustworthy and untrustworthy as well as semi-trustworthy partners. What should say the United States do when there is a need to share data between us and a partner who we believe is untrustworthy, but is still part of our coalition to fight the global war on terror? In our previous paper [17] on Assured Information Sharing,

Proceedings of the 11th IEEE International Workshop
on Future Trends of Distributed Computing Systems (FTDCS'07)
0-7695-2810-4/07 $20.00 © 2007 IEEE

we have discussed the various pros and cons on the need to share model for data sharing.

This paper presents a system to sharing information that is applicable to large teams [1]. A key to the solution is imposing a static network topology on the members of the team where each agent requiring communication to be only along very few links in that network. The key observation underlying this solution is that each piece of information is interrelated and the sender of a piece of information can “guess" who might need some information based on previously sent messages. Thus, when an agent has a piece of information, it can determine which of its neighbors in the network is most likely to either need the information or know who does, based on related messages previously received. Secondly, investigate the influence of different types of team network topology on the efficiency of information sharing.

Trust negotiation is a very important part of any system or an organization. Without trust no transaction can be successful. If there are many systems interacting between them each one has to have trust with other in order to share data, alliances and deals to save the operation cost which is major part of any project. The negotiation is always conflicting since it is to compromise between two agents in order to achieve decision for conflicting distributed systems. The negotiation is taken based on the environment with two decisions to support self interest or the entire system. The decision tree is then formed based on the negotiation and the scenario is stored into the library incase if it is newly proposed. So that it can be used in the future without much of computation.

The Confidentiality of Information is a major threat in a system that is used to share information. In case the confidential information is disclosed to an agent that is not entitled to that level of security, there is a possibility of losing the vital information to an untrustworthy agent. If the trust level of the agent does not match with the security level of the information then the information is secured. The security policy of the information is distinguished into four types as D1, D2, D3 and D4. We call them domains. For more details on security policies, we refer to [18].

The organization of this paper is as follows. Preliminaries are given in section 2. Background and related work are given in section 3. System architecture is described in section 4. Implementation details are given in section 5. Experimental results are given in section 6. The paper is concluded in section 7.

2. Preliminaries

2.1. Definitions

Agent: An agent defines a person or an organization that interacts with other person or organization on behalf of the owner.

Software Agent: It is not as simple as a real world agent. There are various definitions for a software agent. The closest definition would be the following “A software agent is a software with some inbuilt functionalities that interacts with other software agents and perform the allocated task based on the rules that govern them.”

Intelligent Software Agent: It is a hybrid version of a software agent with some intelligence of its own. “[An Intelligent Software agent is] a piece of software that performs a given task using information gleaned from its environment to act in a suitable manner so as to complete the task successfully. The software should be able to adapt itself based on changes occurring in its environment, so that a change in circumstances will still yield the intended result.” (Herman’s 1997)

2.2. Functions

Intelligent software agents should perform the following tasks continuously

1. Insight of changing environment
2. Action required for the change
3. Reason to the action taken
4. Solution for the problem
5. Draw Inferences and perform decision tree for future use.

3. Background and Related Work

Information sharing and Trust negotiation in intelligent agents have there root way behind from 90’s. There are various researches going on Information sharing in Intelligent Software Agents lab of Carnegie Mellon (the Robotic Institute). One of such is Information sharing in Agents. They have alternative decision making systems and Bilateral Negotiations with outside options. In this paper for knowing the background of trust negotiation, will discuss some of the points from the bilateral negotiation with outside options.

The bilateral negotiations paper considers each trust negotiation as a thread. The model is composed of three modules: single-threaded negotiations synchronized multi-threaded negotiations, and dynamic multi-threaded negotiations. The single-threaded negotiation model provides negotiation strategies without specifically considering outside options. The model of synchronized multi-threaded negotiations builds on the single-threaded negotiation model and considers the presence of concurrently existing outside options. The model of dynamic multi-threaded negotiations expands the synchronized multithreaded model by considering the uncertain outside options that may come dynamically in the future.

Most related work can be classified into one of several major categories. The first strand of research is based on a centralized model or distributed model where there are agents such as match maker, information broker or message broad who can response for all information communication [2, 3]. These works has been shown to be able to greatly improve multi-agent [4] system performance [5]. However, such work is inadequate for large team, since it is impossible or undesirable for all team members to share all their information all the time, i.e. because of the limit of required communication channels. The second major strand of research is relies on agents maintaining a shared model of each other or knowing exactly other members' actual internal state as STEAM[6], COM-MTDP [7] and CAST [8]'s mental model. However, as for centralized approaches, in large team there is insufficient bandwidth to support such an approach.

The information sharing problem can also be handled by setting up decentralized model. Both [9] and [10] did a communication decision model based on Markov decision processes (MDP). Their basic idea is an explicit communication action will incur a cost and they supposed the global reward function of the agent team and the communication cost and reward are known. Moreover, [11] put forward a decentralized collaborative multi-agent communication model and mechanism design based on MDP which assumed that agents are full-synchronized when they start operating, but no specific optimal algorithm was presented. Unfortunately, there are no experimental results showing that their algorithm can work on large teams. Incomplete information theory is another way to solve the information sharing problems. [12] Presents a framework for team coordination under incomplete information based on the incomplete information game theory that agents can learn and share their estimates with each other. [13] Used a probability method to coordinate agent team without explicit communication by observing teammates' action and coordinating their activities via individual and group plan inference. Research on social networks began in physics [14, 15, 16], but since it has been applied in many areas though rarely in multi-agent work.

4. System Architecture

The system model for information sharing among large teams can perform distributed information sharing without the cost of maintaining accurate models of all the teammates. First, impose a network topology on the team members analogous to the social networks that exist in human societies. The key characteristic of this network model is that information exchange is based on peer to peer communication. Specifically limit agents to communicating directly with only a small percentage of the overall team.

Leveraging the team network, our basic approach like Figure1 is when an agent has a piece of information to communicate, it forwards that information to the direct acquaintance most likely to actually need that information or know who will. Then the acquaintance performs the same reasoning when it gets the information. After passing through hopefully, a small number of team members, information arrive at a team member that needs it. The intuition is that each agent attempts to guess which of its acquaintance either require the information or are in the best position to get the information to the agent that requires it. Even though members of large teams will not have accurate, up-to-date models of the team, our hypothesis is that the models will be accurate enough to deliver the information in a small number of “hops”. One agent is randomly chosen as the source of some information and another is randomly chosen as the sink for that

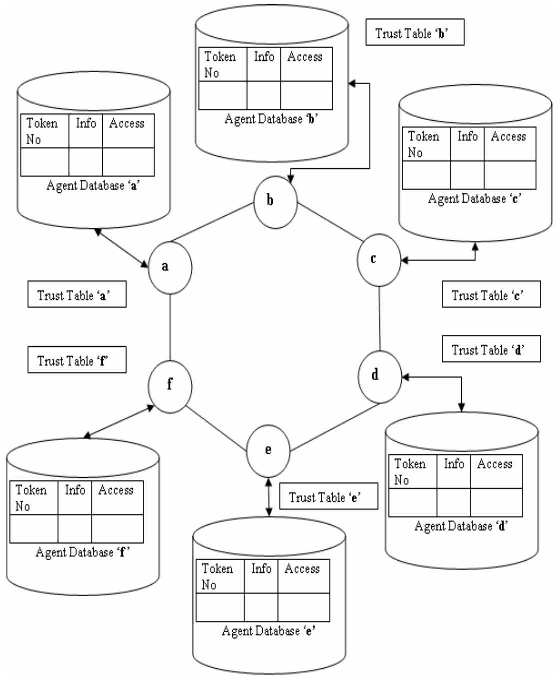

Figure1. System Model for agent information sharing with confidentiality and trust negotiation

information. A probability is attached to each link, indicating the chance that passing information down that link will get the information through the smallest number of links to the sink. The probability will increase as it reaches closer to the target. The chance of missing the target depends on the distance between the source and the sink. The number of "hops" to vary as the distance varies. The challenge is to construct complex models for information sharing but only have reasonable models to improve agent's guessing. The key question is how to create models that allow the agent to "guess" correctly more often than not. To achieve this, we observe that each piece of domain knowledge is typically related to each other piece of domain information. For example, if agent '*a'* tells agent *b* about a plan to destroy an enemy base, when agent *b* gets the information that the base is fake, sending that information to agent *a* is a reasonable thing to do, since *a* likely either needs the information or knows who does. So it is reasonable to infer from an agent's formerly sent message that it may need the other kind of information to improve the performance as the above example. Thus, the previously received information can be interpreted as evidence to infer which acquaintance to send other information to. If an agent maintains a knowledge base about what it heard from its acquaintances, it can use that knowledge to determine where to route newly received information. The other challenge in the network is trust management. Consider the previous example. In case if agent *a* is not trustworthy then that information to destroy the enemy base might be fake. So trust negotiation is an important goal. In our system we can negotiate trust based on their acquaintance. For instance the source is acquainted with another agent in the network that is acquainted with the sink; the sink can get the trust level from its acquaintance. In the beginning the complex network will be formed with no acquaintances. Then once the connection is setup and each agent begin to acknowledge each other's neighbors then the trust levels are assigned to the agent based on their information. If there is a bad agent then it tends to spoil the entire system. The other agent sends the bad acquaintance that they have had with the corresponding agent.

In the system simulation there is also a security policy implementation that has a very important part in the sharing of the information to authorized agents rather than transferring the D4 level data to lower access agents. The token and the information are linked with a security level. Each agent maintains its own level of confidentiality for any particular information. There may be instances where the same information with different clearance domain can be stored in different agents. This also makes a possibility that if one agent rejects the request based on the trust level of the requesting agent, another agent can service the request based on the trust level or acquaintance level that it has maintained with for that corresponding agent. The following example can explain the point. Agent '*a*' can have two or more acquaintances in this case it is two '*b*' and '*c*'. The trust level of '*a*' with '*b*' is in higher clearance domain say D3 and with '*c*' it is in D2. If there is a request from '*a*' sent for some information at D3 then '*c*' will reject the request and '*b*' will service the request. Similarly '*b*' and '*c*' have two different levels for the same information *i.e.* information '*x*' at D3 in '*c*' and at D2 in '*b*'. If '*a*' request for the same information then there is a chance that '*b'* will service the request.

5. Implementation of the System

5.1. Overview

The simulation of the intelligent agents sharing information is done using Java programming. The program mainly concentrates on two things. How much message is being transferred from each agent and the trust element within each agent? The summary of the simulation mainly has results on how much message each agent had in the beginning of the session? How much they shared with the other agents in the simulation and how much they received from the simulation. The important feature of the simulation is that it also holds the history of the summary which makes easy to know the amount of data lost in each session. The agents can make use of the history of the summary to learn more about the other agents in the simulation and try to avoid the more data loss in the future session with the same set of agents. This also helps in knowing the nature of the agents involved, if they are ready to participate and send more messages or they are just waiting to get the most out of the other agents. Such agents are also blocked from the simulation by not sending messages to that particular link. This depends on the individual discretion of the agents. They also pass on the information to other agents in the simulation that such a neighbor is not willing to send any information and readily accepts all the information that is passed on to it or through it. Those dormant agents are like leeches that spoil the entire network.

The agents in the simulation share the information upon request from any other agent in the network. The information with all the agents is inter-related. The messages are numbered in order so as to know the entire flow of the information. Each agent starts collecting the information from other agents based on the information that it has in hand, for example if an agent has a message and its part number 5. It doesn't have any other message numbered prior to 5 or after 5. So the agent first requests for 6 and 4. If it acquires 6 it sends out 7, if it gets 4 the next request is for 3. The agents continue the above way of request until they get whatever they needed from the entire information. The algorithm is explained clearly in the next section of Implementation.

5.2. Algorithm for Information sharing with Confidentiality and Trustworthy Computing

In this algorithm as in Figure 2, at the time of forming the coalition, the agents have the information about the direct acquaintances i.e. a neighbor and their trust level. If there is going to be a new neighbor the trust level is set to a minimum acquaintance level. Then each agent has its own set of information to be shared with other agents in the network. The information is linked with a significant token number and a security Policy. The moment a message is requested by some agent for some information, the token is received then the security policy of the corresponding information is matched with the clearance domain of the requesting agent. The clearance domain mainly depends on the trust level of the requesting agent that is linked with the source agent. In this algorithm there are four such clearance domains: D1, D2, D3, D4 and we assume for simplicity D1< D2 < D3 < D4. The trust levels are similarly split into four levels where in the minimum threshold is set for D1 information (that is information in domain D1). We assume that each agent can read information at all domains, however the trust level that one agent has on the other will determine the domain information that an agent sends to another agent.

There may be multiple copies of the information existing simultaneously in the network along with the same token number, yet the token and information pair is always unique. If the agent gets the same information with two different tokens or vice versa, then his discrepancy will lead to loss of trust. It will perform a multiplicative decrease in the trust level. Similarly if new information arrives trust level of the acquaintance is increased. There is a minimum and maximum threshold level for trust. If any acquaintance falls below the minimum threshold of the trust, then they are removed from the circle of trust, further communication is stopped and the rest of the acquaintances are notified about the bad agent. If the acquaintance's trust level goes above the maximum threshold then the agent sends all the messages requested by the acquaintance. The information sharing goes on until one agent gets the entire information it needs or to a fixed number of time where all the agents have the list of data lost and data gained. It also stores the history of the direct acquaintance and its trust level which helps in future coalition with the same agent.

5.3. Specifications of Algorithm

1. Form Communication link with other agents where the neighbors are the acquaintances.

2. If new neighbor set minimal trust level else load the existing trust level from the database.

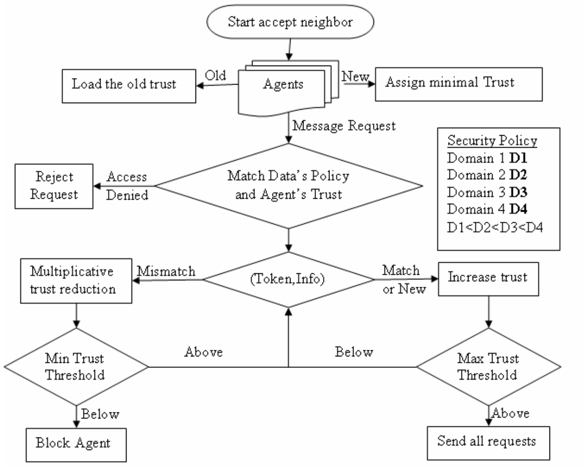

Figure.2 Flow diagram of the Information sharing with confidentiality and trust management.

3. If an agent request for some information. Check the trust level of the agent and the access or security level linked with the information

4. If the access is granted allow service the request based on priority. Else reject request.

5. Start sending and receiving messages (the tokens and the Information are linked).

6. If there is mismatch in messages multiplicative decrease of trust and if the trust goes below minimal trust after decreasing block agent and notify the network

7. If there is message (new or old with match) additive increase trust and also if the trust is above max threshold send the entire request one by one.

8. If any one agent has all information or end of session occurs end link store trust level, Message (Token and Information).

9. Calculate the amount of data lost or gained from *each acquaintance*

6. Experimental Results

The simulation of the algorithm was implemented and there were many sets of results generated. The experimental results were very much helpful in understanding how the system works. In the below chart 1 the Information that was sent from each agent and the information gathered at each end is collected and the Net Gain is also calculated.

Let *T→ Net Gain/Loss of Information for any agent.*

R→The message received from Agents by some agent a_i.

S→ The message sent to other Agents (a_0, a_1*…… *a_n*) by agent* a_i.

O→ The own message of each agent in the beginning of the session.

T = (R-(S+O))

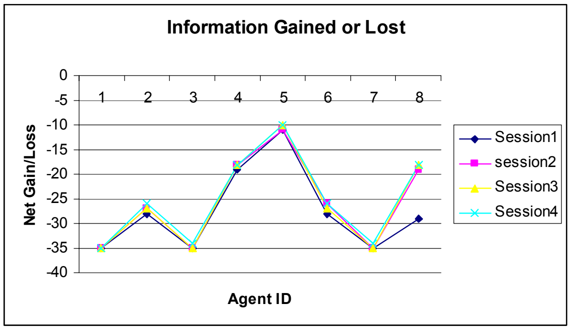

Chart 1. Net gain of information by each agent in four continuous sessions of information sharing with trust computing.

The Chart 1 has four set of simulations that was done within 8 agents. The simulation revealed that as the session increases the gain also increases. This is because the agents come to know well about the other agents. Agents have the trust level of each agent in their database summary. The trust level increases the gain increases. Since most of the agents send 90% non negative messages the gain increases for each session. Gain is the amount information collected from other agents apart from the ones it has forwarded to other agents without storing it in its database, for example if an agent has 10 messages with it before the start of the session, it receives 25 messages from other agents of which its request is 15 messages. The number of messages sent by the corresponding agent to other agents in the session in response to their request is considered as loss of information say 5. The net gain is (25-(15+5)) is equal to 5. The chart1 also clearly shows that there is not a much of difference in gain with each agent in successive session. They all share same level of trust in the beginning and the gain varies based on their trust level through the simulation. If they send one negative message their gain goes down. The neighbors stop sending messages if they are notified that some agent is below the threshold level of some other acquaintance. So the gain in sharing depends mainly on the trust level. The trust of the node is directly proportional to the quality of the message sent by that agent and the gain is also directly proportional to the trust achieved by the agents. The chart has net gain and loss on its Y axis. The series one to four indicate the simulation that was conducted on the eight agents in continuous session of information sharing. The chart clearly indicates the increase in gain as the session progresses. The level of gain varies from one agent to another because of the neighbors, amount of information the neighbors possess with them. For instance in the chart agent 5 has its gain level in -10 and rest of the agents have it in -25 to -35. The reason is due to the neighbors of the agents are 4 and 8. So the information provided by the neighbors gives more gin to agent 5 than other agents. 4 and 8 also in turn get more gain from 5 and stand higher than the other agents. The summary of the experimental results contain the amount of message sent, received and the Net. It also has the recent trust level of all the neighbors. The newly received tokens are also copied in the summary.

7. Summary and Future Directions

The proposed algorithm has been implemented. The experimental results show that the information sharing is done as in peer to peer communication network. The amount of information lost and gained is stored at the end in the database. The number of messages sent to share a little amount of information through the network is high. The scalability also becomes an issue. If there are more neighbors the amount of message sent and managing the traffic of messages becomes a very big issue. The future work on this research can be implementation of the above system in which the guess and hops are calculated to the efficient way to share information among the agents.

A major issue we leave for future research is how to calculate the relationships between pieces of information which is highly relative with domain knowledge and expertise where our algorithm should be applied. Furthermore, we do not investigate how information sharing works on negative relative messages where the relationship between pieces of information. Does the dormant agent gain more than the other active agents? Can the agents form a multicasting group which might help in communicating with a group of agents simultaneously? The multicasting group will save a lot of network resources by sending one message to a gateway agent and thereby pass it to the whole multicast group.

Acknowledgements

The work reported in this paper is supported by AFOSR under contract FA9550-06-1-0045 and by the Texas Enterprise Funds. We thank Dr. Robert Herklotz for funding our research on Information Operations Across Infospheres. We thank Profs. Latifur Khan, Murat Kantarcioglu, Elisa Bertino, Ravi Sandhu and Tim Finin as well as Dr. Mamoun Awad and Dr. Ebru Celikel and many others for discussions and inputs on Assured Information Sharing (AIS). We also thank the students Sai Chaitanya, Abinandhan Chandrasekaran, Ryan Layfield, Nathalie Tsybulnik, Li Liu, Alam Ashraful, Ganesh Subbiah, Gal Lavee, Kim Jungin, and Pavan Chitumalla as well as many others for discussions on AIS. This paper will be published in the Proceedings of the 2007 IEEE Workshop on Future Generation Distributed Computing Systems.

References

[1] P. Scerri, Y. Xu, E. Liao, J. Lai, M. Lewis, K. Sycara. Coordinating very large groups of wide area search munitions, Recent Developments in Cooperative Control and Optimization, Dordrecht, NL: Kluwer Academic Publishers.

[2] M. H. Burstein and D. E. Diller. A framework for dynamic information flow in mixed-initiative human/agent organizations. Applied Intelligence on Agents and Process Management, 2004. Forthcoming.

[3] K. Decker, K. Sycara, A. Pannu and M. Williamson. Designing behaviors for information agents. Procs. Of the First International Conference on Autonomous Agents, Feb., 1997.

[4] P. R. Cohen, H. J. Levesque and I. Smith. On team formation.In J. Hintikka and R. Tuomela, editors, Contemporary Action Theory, Synthese, 1998.

[5] K. C. Jim and C.L. Giles. How communication can improve the performance of multi-agent systems. In Proceedings of Autonomous agents'01, 584-591, 2001.

[6] P. Scerri, Y. Xu, E. Liao, J. Lai, K. Sycara. Scaling Teamwork to Very Large Teams, AAMAS 04, Forthcoming, 2004.

[7] D. Pynadath and M. Tambe. The communicative multiagent team decision problem: analyzing teamwork theories and models. Journal of Artificial Intelligence Research, Vol.16, pages 389-423, 2002.

[8] J. Yen, J. Yin, T. R. Ioerger, M. S. Miller, D. Xu and R. A. Volz. Cast: Collaborative agents for simulating teamwork. In Proceedings of IJCAI'01, pages 1135-1142, 2001.

[9] P. Xuan, V. Lesser and S. Zilberstein. Communication decisions in multiagent cooperation: Model and experiments. In Proceedings of Autonomous Agents'01, 2001.

[10] C.V. Goldman and S. Zilberstein. Optimizing information exchange in cooperative multi-agent systems. Proceedings of the Second International Conference on Autonomous Agents and Multi-agent Systems, 2003.

[11] C.V. Goldman and S. Zilberstein. Mechanism design for communication in cooperative systems. Game Theoretic and Decision Theoretic Agents Workshop at AAMAS' 03, July, 2003.

[12] H.H. Bui, S. Venkatesh and D. Kieronska. A framework for coordination and learning among team members. In Proceedings of the Third Australian Workshop on Distributed AI (DAI-97), pages 116-126, Perth, Australia.

[13] M.V. Wie. A probabilistic method for team plan formation without communication. Proceedings of the Fourth International Conference on Autonomous Agents, pages 112-113, Barcelona, Spain, June 3-7, 2000.

[14] R. Albert and A. Barabasi. Statistical mechanics of complex networks. Review Modern Physics, 74, 47,2002.

[15] M. E. J. Newman. The structure and function of complex networks. SIAM Review, Vol. 45, No. 2, pages 167-256, 2003.

[16] D. Watts and S. Strogatz. Collective dynamics of small world networks. Nature, 393:440-442, 1998.

[17] B. Thuraisingham, Assured Information Sharing, UTD Technical Report, December 2006 (also to appear as Book Chapter in Data Mining for Security Applications edited by H. Chen et al)

[18]. B. Thuraisingham, Database and Applications Security, Integrating Data Management and Information Security, CRC Press, 2005.