An Agent Based Reputation System for Unreliable Grid Environment

Punam Bedi, Harmeet Kaur, and Bhavna Gupta

***Abstract*—To make grid computing environment a robust platform, selection of resource owners by users for the job delegation and the protection of owners' resources from malicious users must be carefully handled. In smaller grids due to possession of identity certificates by the users, the above problem was not prominent. But as the grids grow larger and jobs become more complex, the traditional methods cannot be used due to lack of flexibility and scalability in them. In this paper we propose an agent based reputation system to solve the above problem of selection of resource owners for the job delegation and users for the security of resources. Our framework computes trust and reputation among interacting grid entities based on community experiences to classify and select them for interaction. To quantify trust and reputation of the grid entities Fuzzy Inference system is applied.**

***Index Terms*—Agent, fuzzy inference engine, recommendation, trust.**

I. Introduction

The fundamental idea of grid was resource sharing. In past, due to local small grids, the partners of the grid knew each other so they used to collaborate to complete the experiments. But as the grid usage for commercial purposes is realized, resources get shared among many of the unknown parties and there is a requirement that interacting parties need some measure for authentication and validation for each other. Traditional methods of authentication like digital certificates works fine for small size grids where number of nodes is small. But as the grid grows larger and as jobs become complex, authentication and authorization becomes hard to be provided with traditional methods because of the requirement of possession of a large set of identity certificates by the users as well as by the resource owners as a single job may access resources with many different owners with different requirements. So there is a need of a mechanism in large grids by which selection of resources by users and checking the validity of users by resource owners can be done. As it happens in societies wherever selection of the things is required, trust plays a major role. By employing trust factor in decisions, risk gets reduced. In grid also to improve the performance, trust can be employed i.e. user can submit the task to only trustworthy reputed resource owners so that the risk of failure of task is low and resource owner will provide the resources only to trustworthy reputed users to protect their resources.

Manuscript received October 9, 2012; revised December 20, 2012.

Punam Bedi is with Department of Computer Science, University of Delhi – 110007, India (e-mail: punambedi@ieee.org).

Harmeet Kaur is with Department of Computer Science, HansRajCollege, University of Delhi – 110007, Delhi, India (e-mail: hkaur@hrc.du.ac.in).

Bhavna Gupta is with the Department of Computer Science, University of Delhi, India (e-mail: guptac7@gmail.com).

In this paper we are proposing an agent based reputation system for grid where the grid is considered as a multi agent system in which every entity is accompanied by an agent whether it is a user or a resource owner. Users of the grid submit their task with explicit requirements to their corresponding agents. It is now the responsibility of a user agent to select the appropriate resource owner for a task given by the user. For selection, each agent uses its database where besides its personal experience; a list of its trustworthy acquaintances with their trust factor is maintained. Whenever there is a request of selection of a resource provider, user agent prepares a query i.e. a request vector based on the requirements of the user. This query is passed to its trustworthy agents to get the recommendation. This process goes on till the timeout variable permits. The timeout variable stores the time period for which a user can wait to have the result of request. Initial value to timeout variable is specified by the user at the time of specifying other requirements.

Here in our system recommendations are collected from trustworthy acquaintances as a set of parameters like cost, timeliness, response time, security provided to the job, availability of the resources etc. which help in computing the reputation of resource providers. To compute the reputation from these aggregated parameters fuzzy inference system is used because different membership functions and inference rules could be developed for different grid applications, without changing the fuzzy inference engine and because of its capability of quantifying the uncertainty present in various aggregated parameters as recommended by the trustworthy acquaintances. So using the reputation values as computed by the fuzzy inference engine about various resource providers, a list is prepared by the user agent to decide upon a resource provider.

Once the user agent prepares a list of resource providers on the basis of reputation value, it is now resource agents turn to compute the reputation of the user who want to use the resources. For that the user agent submits the profile of the user to the selected reputed resource provider's agents present in the list. These agents then compute the reputation of that user and if both parties agree on reputation factor then only trustworthy user can delegate the task to trustworthy resource owner with minimum risk involved in transaction. Once the transaction is over, agents update the trust on each other for future recommendations. The paper is organized as follows. Section 2 describes our frame work. Section 3 details related work. Finally a case study and conclusion is in section 4 and 5, respectively.

II. Agent Based Reputation System for Grid (The Framework)

Two major challenges faced by market oriented grid are as follows

- User applications may contain malicious code which can harm resources.
- The shared grid resources may contain the malicious code to harm the user application.

The above problems are seen because in large sized grids, grid entities do not have knowledge about each other. Due to these limitations grid applications are prone to failures. So there is a need of a mechanism by which these failure rates of the jobs can be reduced. As it happens in real life, people prefer to interact with those who have trustworthy reputation. So if somehow reputation of the grid users and the resource providers can be computed, the grid interactions will be more reliable. The definition of the reputation taken in this paper is as follows "Reputation is a value corresponding to each of the entity present in the grid environment based on the trust exhibited by it in the past" [1] i.e. if a resource provider is known to provide certain good qualities of resources over a period of time irrespective of its limitations, then it is assumed to have a good reputation.

As in a grid environment resources are geographically distributed and owned by the different individuals so it is suggested to take multi agent environment in the proposed framework [2]. The multi agent infrastructure proposed is organized in such a way that every entity in the grid is accompanied by an agent. In this proposed framework, there are two types of agents:

1) User agent: Users of the grid submit the task to their corresponding user agent specifying requirements e.g. workload, execution deadline, budget limit, minimum reputation of the resource provider etc. It is now the responsibility of the user agent to select a suitable resource provider for the user satisfying all the requirements specified by her/him. Here in proposed system, the user agents act as a community and help each other at the time of request of any user's query. Every agent maintains a database in which besides their personal experience about resource providers, trust on various trustworthy agents is also stored.
2) Resource provider agent: Every resource owner communicates with the grid through his agent called resource provider agent. The data corresponding to every resource such as its availability, its workload is maintained by this agent.

Basic methodology of the system consists of the following steps:

1) Generation of a query from the requirements of the grid user.
2) Generating recommendations for a user.
3) Updating trust level on trustworthy acquaintances

A. Generation of a Query from the Requirements of the User

As in development of any project requirement specification phase is very crucial, so is this step. User of the grid gives his requirements to his agent, known as the user agent. The requirements can be given as execution deadline, budget limit, minimum reputation of the resource provider etc. Prior job execution success rates, job turnaround time are many other factors are there which a user can specify, if required, in his service demand.

Once the specification of the requirements is given to the agent, the user agent will prepare a query in the form of request vector from those specifications generating an identity number to the query and sends this vector with identification number to its trustworthy agents to get their recommendation.

B. Generating the Recommendations for a User

In our framework agents exist in a network that acts as a society and follow social recommendation process. As we know in our society people generally use their own experience and take recommendation from their trustworthy acquaintances which act as referral recommendations. For example if A trusts B and B trusts C and if A requests something from his trustworthy acquaintance B, B can also take help from his trustworthy acquaintance C to give recommendations. So finally A has referral recommendations from those persons who are unknown to him i.e. C. Similarly in our system at the time of request, the user agent take recommendation from those known to it, they may further take recommendations from those known to them and so on. Let x_{ij} represents the trust of agent i on agent j and xji represents trust level of agent j on agent i. Also it is not necessary that $x_{ij} = x_{ji}$. "fig1" depicts this terminology i.e. if x_{12} represents trust level of agent 1 on agent 2 and x_{21} represents trust level of agent 2 on agent 1 and it is not necessary that $x_{12} = x_{21}$.

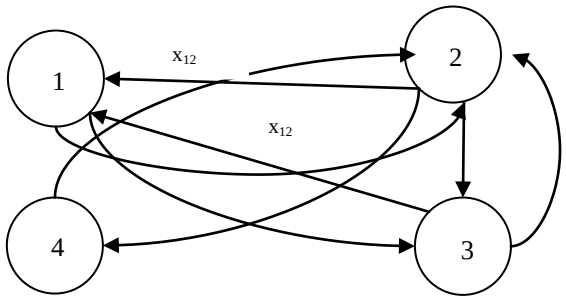

Fig. 1. Web of Trust among Agents

These trustworthy relationships form a web of trust and the advantage of this web is that agents are able to get the recommendations from even those agents that are not known to it i.e. agent 1 can get recommendations from agent 4 also. Once a request is obtained from a user agent, there are two ways that a trustworthy agent can generate recommendations [3]:

1) The agent has interacted directly with the resource provider.
2) The agent comes to know about a resource provider through his trustworthy acquaintances.

If the agent has already used the services of a resource provider then that resource provider will be a part of the recommendation list only after comparing expertise vector of the resource provider and the request vector [4]. The expertise vector of a resource provider is prepared based on the attributes of the services he is providing. The user agent query can also be translated into request vector with the same dimension as of expert vector. After finding the cosine similarity between these two vectors, agent can conclude

whether a resource provider is capable to provide services to requesting user or not. Capability of a resource provider can be calculated as in "(1)"

$$C = R \otimes E = \frac{\sum_{i=1}^{n} (r_i e_i)}{\sqrt[n]{\sum_{i=1}^{n} r_i^2}} \quad (1)$$

where R (<$r_1 \ldots r_n$>) is a request vector and E(<$e_1 \ldots e_n$>) is an expertise vector and n is the number of dimensions these vectors have. To calculate $r_i e_i$ for parameters which are not binary, significance of the parameters is to be considered [5]. For example if requesting user can spend maximum (r_i) of Rs.3500/- to get a resource and resource provider is providing the service @ (e_i) Rs.2500/- then $r_i e_i = 1$ otherwise 0 i.e. if $r_i >= e_i$, then $r_i e_i = 1$ otherwise 0 whereas in case of prior job execution success rate of the resource provider, if the requesting useragent wants that the resource provider must have at least 60per cent success (r_i) and the resource provider has 80per cent(e_i) then $r_i e_i = 1$ otherwise 0 i.e. if $r_i <= e_i$, then $r_i e_i = 1$ otherwise 0.

If $C > C_{\min}$, where $C_{\min}$ is the minimum value of the capability of the resource provider desired to perform the job, then further aggregation of his service attributes with those recommended ones is done and then passed to the requesting agent. The aggregation of the service attributes (parameters)as recommended by the trustworthy acquaintances is done as follows. Let P_1, P_2 …. P_m (scale between 0 – 9) be the m parameters to be aggregated. Let x be the agent who is aggregating these parameters obtained from his n trustworthy acquaintances i.e. Y_1, Y_2, Y_3…. Y_n are the trustworthy acquaintances with their respective trust values as t_1, t_2, t_3 ……t_n. Then aggregation of m parameters is done by the agent x using "(2)"as

$$\text{Agg}(P_{ix}) = \alpha P_{ix} + \beta(t_1 \times P_{i1} + t_2 \times P_{i2} + t_3 \times P_{i3} + \ldots.. + t_n \times P_{in}) \quad (2)$$

where α, β are the weights assigned to the information collected using direct and the indirect experience and $\alpha + \beta = 1$. P_{ix} represents the value of i^{th} parameter stored in the database of agent x as its direct experience whereas P_{i1}, P_{i2}, …, P_{in} represents the value of i^{th} parameters as recommended by trustworthy agent 1, agent 2, …, agent n with trust value t_1, t_2, …,t_n, respectively.

The process of taking recommendations from trustworthy agents and doing aggregation of the parameters goes on further till timeout variable permits. Timeout variable is given a value say t by user depending upon the time interval for which a user can wait to get result. This variable gets decremented with system clock. Before further sending the request vector to trustworthy acquaintances, to get recommendation, every agent checks timeout variable value which tells in how much time he has to give recommendations to requesting agent. If time is there to get the recommendation from his trustworthy acquaintances, then further sending of the request vector takes place otherwise agent responds back with a list of aggregated variables of resource providers based on his knowledge to requesting agent.

The aggregated parameters are in the scale of (0-9) and are actually the representation of a linguistic terms e.g. if recommendation for a resource provider's prior job execution success rate attribute is asked from a trustworthy acquaintances then instead of saying "very good" or "poor", it is passed as a number between 0 to 9 which helps in applying the fuzzy inference engine. To infer the uncertainty present in the recommended parameters, each of the attribute chosen is considered as a fuzzy variable and characterized by a membership functions. These membership functions of recommended aggregated parameters are used by the requesting agent to compute the reputation of the resource providers by applying the fuzzy inference system. There are certain advantages of applying fuzzy inference system in grid application because of its capability of handling uncertainty and impreciseness present in the recommended parameters about resource providers and also different membership functions and inference rules could be developed for handling different grid applications without changing the fuzzy inference engine.

A standard fuzzy inference process consists of major four steps. The computation of reputation can be done by applying those steps is written in algorithm 1below:

Algorithm: Fuzzy inference procedure to compute reputation

1) Find the membership function of every aggregated parameter i.e. job turnaround time and job execution success rate.
2) Apply the fuzzy rule set to map those aggregated input parameters space to output reputation space.
3) Aggregate the fuzzy outputs of all the above rules written to a fuzzy set i.e. reputation.
4) Compute numeric crisp value of reputation with defuzzification of the fuzzy output.

In step 3 for aggregation among many methods, root sum square method is used in the paper. To apply that firing strength of each fuzzy rule is calculated beforehand using the formula given in "(3)".

$$P_i = \sqrt{\sum_{j=1}^{k} \sum_{i=1}^{n} d_{ij}^2} \quad (3)$$

where P_i represents the firing strength of the rule R_i used to compute fuzzy variable f_i and d_{ij} represents the degree of membership of the input variable x_i to the rule R_i. Here n represents the number of input variables and k represents the number of rules that gives output as fuzzy variable f_i. The aggregated output as computed in step 3 when defuzzified gives us a crisp value i.e. reputation of a resource provider and for defuzzification the formula used in step 4 is given in "(4)"

$$reputation = \frac{\sum_{i=1}^{n} C_i P_i}{\sum_{i=1}^{n} P_i} \quad (4)$$

where C_i denotes the centroid of the trapezoid i^{th} fuzzy variable and P_i denotes the firing strength of the i^{th} fuzzy variable.

Once the reputation of all the recommended resource providers are computed by the above method then the user agent prepares a list of resource providers keeping in consideration the minimum reputation as specified by the user for job delegation. Then the user agent sends the user profile to all selected resource providers present in the list. After getting the profile of the user, resource providers' agents now compute the reputation of the user. Normally user reputation is computed from his past performance when using certain services under specific contract, reflects to what degree the requester can be expected to fulfill its declarations in a service contract. If resource provider agent finds reputation of the user suitable then it responds back with terms and conditions to pursue the job. If the user also agrees for those terms and conditions finally the job is delegated to the communicating reputed resource provider with reduced risk of failure of the job.

C. Updating trust level on trustworthy acquaintances

The user agent gets the recommendations from various trustworthy agents and used them for the delegation of the job. Once the delegation of the job is done the user agent has to update the trust on recommenders. The updation is done on taking in account the difference in reputation values as obtained by aggregating the recommendations and as given by an agent.

Depending upon whether the difference is below a threshold d_t, the user agent updates the trust value on recommenders as follows in "(5)"

$$\text{Trust}(R) = \text{Trust}(R) + (d_t - d) \quad (5)$$

where *d* represents the difference between the aggregated value of recommendations at the user agent and the individual recommendation of the recommender.

Hence if there is agreement, $(d_t–d)$ is positive then trust level on trustworthy acquaintance will increase but if there is difference in opinions then trust will decrease. This update process of the trust level is a continuous process and is done at every agent as and when the delegation of the task is done.

III. Related Work

Reputation is not a new concept for online community, for e-commerce and for multi agent systems. In grid systems, usually reputation and trust are constructed and maintained for security reasons [6] for e.g. a security aware model between resource provider and the consumers is presented in [7], [8] also presented a trust enhanced security solution for the enhancement of grid security. The above models provide an improved system but limitation is that their application of trust is only to grid security Then [1] proposed a reputation management framework i.e. Grid Eigen Trust for selecting the resources after computing the reputation of various entities. In [9], a hierarchical model for reputation calculation is proposed. The above system lacks efficiency because it is computing reputation of a grid entity at three different levels i.e. virtual organization, institution and grid entity. The distinctive feature of our work is that fuzzy inference approach is used to compute the reputation among entities. The advantage of using fuzzy inference system is its flexibility i.e. for different grid applications, without changing the main fuzzy inference engine, different inference rules and membership functions could be developed. [10] also used fuzzy inference system but for matching the self-claimed security parameters of the client to the security policy of the Grid server. The work proposed in this paper is not only handling the security issue but also make the transactions to happen only between reliable grid entities with reduced risk of failure of the job by employing trust and reputation factors in selection of grid entities.

IV. Case Study

This case study presents one application for our proposed model where a user wants to submit a job to be executed on an upgraded processer. There are multiple instances of the processor which are lying with different resource owners of the grid as one of the resource. The user instructs its agent to submit the job and get the job done from that resource owner who satisfies his specified requirements. According to proposed model user agent fires the query as a request vector to his trustworthy acquaintances for recommendation about various resource providers. Suppose four trustworthy agents of the user agent responded back with aggregated service attributes of the resource providers satisfying the user requirements as shown in Table I.

Let the data already present with user agent as a direct experience is represented as {(rp1, 4, 40%), (rp2, 6, 10%) (rp3, 2, 80%) }.

TABLE I: Recommendation of Various Agents as Given to User Agent

Agent	Trust Level	Resource provider	Agg. (Job turnaround time	Agg. (Prior job success rate)
Agent 1	0.8	rp1	4	40%
		rp2	6	50%
Agent 2	0.4	rp1	2	40%
Agent 3	0.2	rp1	5	50%
		rp2	6	80%
		rp3	9	90%
Agent 4	0.6	rp2	5	20%

Aggregation of the parameters was done at the user agent using "(2)" with $\alpha = 0.7$ and $\beta = 0.4$ is shown in Table II.

TABLE II: Aggregation of the Recommendation

Resource provider	Agg.(job turnaround time)	Agg. (prior job success rate)
rp1	4.8	5.1
rp2	7.3	3.2
rp3	0.3	0.3

Each of these aggregated parameters are now used to apply fuzzy inference procedure as stated in algorithm. According to step 1 of algorithm each parameter will map to a fuzzy variable and get its membership value according to the membership function as given in Fig. 2 and Fig. 3 for job turnaround time and prior job success rate attribute respectively.

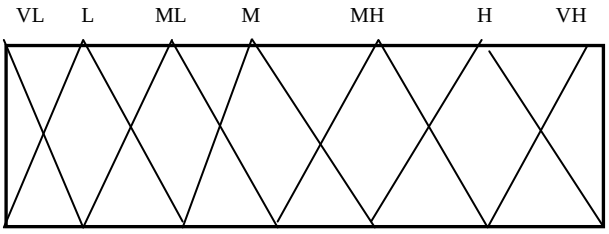

Fig. 2. Membership graph of Job turnaround time

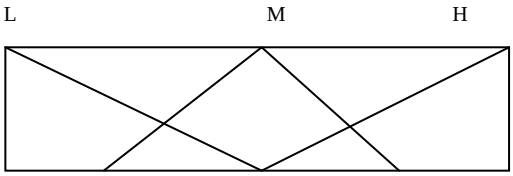

Fig. 3. Membership graph of Prior job success rate

After applying the fuzzy rules according to step 2 of the algorithm on these aggregated parameters mapping of the input space to output space is done. Table III shows the set of fuzzy rules that are designed for this scenario.

TABLE III: SET OF RULES FOR FUZZY INFERENCE ENGINE

Attribute name	Job turnaround time	Prior job success rate	Reputation
Rule R1	High	Medium	High
Rule R2	High	Low	Low
Rule R3	Medium	Medium	Medium

The Fig. 4 shows the complete execution of the algorithm to compute the reputation of one of the resource provider i.e. for (rp1).

Fig. 4. Applying algorithm to compute the reputation of the resource provider

Once the mapping of the input space to output space is done next step is to aggregate the output obtained from each rule and for that the firing strengths for various rules written in Table III using "(3)" are computed as shown in Table IV

TABLE IV: COMPUTATION OF THE FIRING STRENGTH OF FUZZY RULES FOR AGGREGATION

Rules	Root sum square	Firing strength for fuzzy variable
R1(high)	$(9^2 + 5^2)^{1/2}$	10.29
R3(medium)	$(5^2 + 5^2)^{1/2}$	7.07

Using this values defuzzification was done using the formula specified in "4". Therefore the reputation of the rp1 is:

$$\text{Output} = 5.759/17.36 = 0.3317$$

as a crisp output and denotes the rank of the resource provider rp1.

Similarly reputation of the other resource providers were computed with the above stated procedure and shown in the Table V.

TABLE V: REPUTATION OF VARIOUS RESOURCE PROVIDERS

Resource provider	Rank (reputation)
rp1	0.331
rp2	0.1429
rp3	0.307

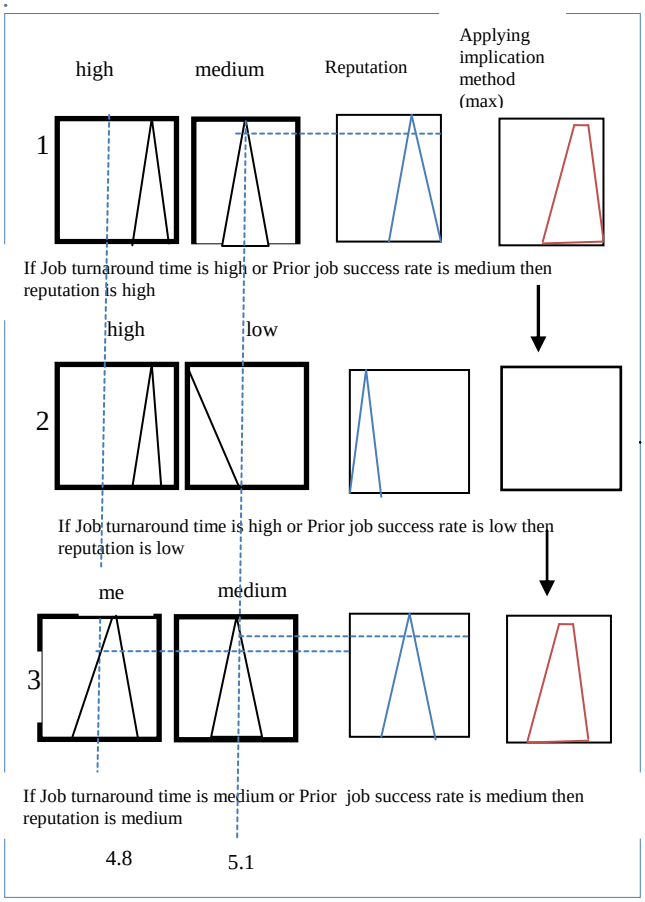

Fig. 4. the complete execution of the algorithm to compute the reputation of one of the resource provider

Comparing the reputation values of the resource providers obtained, job is delegated to the highest reputed resource provider i.e. to rp1 provided the resource provider also satisfies with the reputation of the user

V. CONCLUSIONS

In this paper a method of selecting grid entities for accomplishment of the job is presented by applying fuzzy inference engine. The reputation of various grid entities are computed and interaction is done only between reputed grid entities and thus reducing the overall cost of the accomplishment of job and making the grid environment more robust. Applying fuzzy inference engine to compute the reputation of various interacting grid entities seems logical because of its simplicity and flexibility and also because of its way of balanced and comprehensive decision making mechanism that mimics the human thinking.

REFERENCES

[1] B. Alunkal, I. Veljkovic, G. Laszewski, and K. Amin, "Reputation Based Grid Resource Selection," in *Proc. Workshop on Adaptive Grid Middleware*, 2003.

[2] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke, "Grid Services for Distributed System Integration," *Computer*, vol. 35, issue 6, pp. 37-46, June , 2002

[3] P. Bedi and H. Kaur, "Trust Based Recommender System," in *Proc. Int'l Conference on AI,* 2005, pp. 798-801.

[4] P. Bedi and H. Kaur, "Modelling Agent Reputation in Uncertain Environment," in *Proc. International conference on Cognitive Science,* Allahabad, India, 2004, pp. 258-263.

[5] B. Gupta, H. Kaur, and P. Bedi , "A Reputation based Service provider Selection for Farmers," *Journal of Information Assurance and Security*

(JIAS), vol. 5, pp. 515, 2010.
[6] M. Adamski, A. E. Arenas, and A. Bilas, "Trust and Security in Grids: A State of the Art," *CoreGRID White Papers, WHP-0001* , 2008.
[7] F. Azzedin and M. Maheswaran, "A trust brokering system and its application to resource management in public resource grid," in *Proc. 18th International Parallel and distributed Computing Symposium (IPDPS)*, Santa Fe, IEEE Computer Society Press, 2004, pp. 22 .
[8] G. C. Silaghi, A. E. Arenas, and L. M. Silva, "Reputation-based trust Management Systems and their Applicability to Grids," *CoreGRID Technical Report TR-0064*, Feb. 23, 2006,
[9] I. Foster, C. Kesselman, and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," in *Proc. 7th International Euro-Par Conference*, Aug. 28-31, 2001, pp. 1-4.
[10] S. Song, K. Hwang, and Y. K. Kwok, "Trusted grid computing with security binding and trust integration," *Journal of Grid Computing*, vol. 3, pp. 53-73, 2005.

Punam Bedi received her Ph.D. in Computer Science from the Department of Computer Science, University of Delhi, India in 1999 and her M.Tech. in Computer Science from IIT Delhi, India in 1986. She is an Associate Professor in the Department of Computer Science, University of Delhi. She has about 26 years of teaching and research experience and has published more than 130 research papers in National/International Journals/Conferences. Dr. Bedi is a member of AAAI, ACM, senior member of IEEE, and life member of Computer Society of India. Her research interests include Web Intelligence, Soft Computing, Semantic Web, Multi-agent Systems, Intelligent Information Systems, Intelligent Software Engineering, Software Security, Intelligent User Interfaces, Requirement Engineering, Human Computer Interaction (HCI), Trust, Information Retrieval, Personalization, Steganography and Steganalysis.

Harmeet Kaur was born in 1973. She is currently an Associate professor of Computer Science in Hans Raj College, University Of Delhi, Delhi-110007, India. Her research interests lie in the field of Trust, Recommender Systems, Grid Computing, Cloud Computing and Multiagent environment.

Bhavna Gupta was born in 1974. She is now a research scholar under the supervision of Dr. Punam Bedi and Dr. Harmeet Kaur. Her research fields are Trust, Grid computing, Cloud Computing, Multi agent environment.