http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (12): 2296-2305, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.2296.2305

SLA Based Trust Management for a Pervasive Environment

J. Valarmathi and V. Rhymend Uthariaraj
Computer Centre Ramanujan Computing Centre, Anna University, Chennai, India

Abstract: Service Level Agreements plays a major role in pervasive applications. In pervasive computing, trust Management is an important issue where consumers and providers are distributed geographically across autonomous domains. In reputation based trust models, the communication between provider and consumer is established by computing trust index obtained from neighboring entities. This involves many numbers of transactions. In this paper, using policies both consumer and provider exchange their policy information prior to communication. This work also addresses the establishment of trustworthiness of the entities and proposes SLA based trust model. The proposed model is shown to be robust in terms of selection of suitable trustworthy provider, elimination of false feedback to safeguard provider's trust-index and improve job success rate for the consumer community.

Key words: Trust management, reputation, service level agreement, policies

INTRODUCTION

Trust is the prerequisite for any computing system to allow the devices to access the shared resources and services. Every communication is established based on trust value of an environment. It provides devices with a natural way of judging other devices, similar to how security and privacy are handled in human society. When a trust value of an entity among entities in a community is higher, then there is a great chance of completing the service successfully. Trust builds networks that are durable, flexible and efficient. It provides devices with a natural way of judging other devices, similar to how we have been handling security and privacy in human society. Trust improves the quality of services and also makes a network work successfully.

Pervasive computing is a decentralized open environment (where communications between devices, with minimal or no human intervention, are commonplace) where different devices interact without prior knowledge of each other, involving a large amount of data.

These environments must provide effective security and privacy mechanisms to protect these data and ensure the quality of interactions. Without human judgment, devices need to distinguish other peers' identities and behaviors autonomously in the pervasive computing environment (Traek, 2011). It does not suffice merely to authenticate devices because most are unknown and there usually is no central management mechanism. Pervasive environments are distributed and mobile in nature. Therefore, conventional methods cannot be done for providing security. Most of the conventional methods are processor dependant, which is not suitable for pervasive devices. Pervasive devices are functioned by their low energy. Therefore trustworthy computation is used to provide better solutions.

In most of the conventional trust model, communications between entities are based on reputations. Prior communication (Bussard *et al*., 2004) between consumer and provider is required before providing service which is not possible in the conventional trust models (Shuai *et al*., 2010). This is overcome by the proposed work using service level agreement. The SLA is a legal formal documenting the way that services will be delivered as well as providing a framework for service charges. Service Providers use this foundation to optimize their use of infrastructure to meet signed terms of services. Service consumers use the SLA to ensure the level of quality of service they need and to maintain acceptable business models for the long-term provision of services.

SLA OVERVIEW

A service-level agreement is a part of a service contract where the level of service is formally defined. In practice, the term SLA is sometimes used to refer to the contracted delivery time (of the service) or performance. An SLA is a document that describes the minimum performance criteria a provider promises to meet while delivering a service. It typically also sets out the remedial action and any penalties that will take effect if contract between a service consumer and the provider is violated. Pelta and Yager (2010) describes service-level agreement

Corresponding Author: J. Valarmathi, Computer Centre Ramanujan Computing Centre, Anna University, Chennai, India
Tel: (0) 9884387238 Fax: (044) 22516008

is a negotiated agreement between two parties, where one is the customer and the other is the service Provider. This can be a legally binding formal or an informal "contract". The SLA records a common understanding about services, priorities, responsibilities, guarantees and warranties.

The SLA may specify the levels of availability, serviceability, performance, operation, or other attributes of the service. In some contracts, penalties may be agreed upon in the case of non-compliance of the SLA. Any SLA management strategy considers two well-differentiated phases: the negotiation of the contract and the monitoring of its fulfillment in real-time. Thus, SLA Management encompasses the SLA contract definition: Basic schema with the QoS (quality of service) parameters; SLA negotiation; SLA monitoring and SLA enforcement-according to defined policies. A well defined SLA, typically includes a number of components:

A Description of the provided service nature: This includes the type of the service and a description of the technical issues associated with the service, such as, network connectivity, operation and maintenance; in addition the server's and client's configurations.

The level of responsiveness and reliability of the service: This includes availability requirements and how soon the service performs in a normal state.

Service problems reporting: This includes who will be contacted in case of a certain problem, in addition to steps and formalities that must be followed to guarantee quick resolving of the problem.

Problem response time and resolution: This defines the time after which a reported problem will be solved.

Monitoring and service reporting: This defines how quality levels are monitored and reported, who is responsible of that, how and how often.

Liabilities on the service provider if promises were not met: In such cases, a service customer may be given extra credits, like the customer can have the power to terminate the contract, or ask for refunding.

Extra conditions: These are conditions upon which the SLA is not valid anymore. This can be true, for example, in case of natural causes (e.g., flooding, fire). Also such cases may involve the customer himself trying to breach the security of the service provided or the network.

According to Sun and Denko (2008), an SLA life cycle consists of five phases:

- **SLA Development:** In this phase the SLA templates are developed
- **Negotiation and sales:** In this phase the SLA is negotiated and the contracts are executed
- **Implementation:** Where the SLA is generated
- **Execution:** The SLA is executed, monitored and maintained
- **Assessment:** Evaluation of the SLA performance. In this phase, a reevaluation of the initial SLA template might be done

RELATED WORK

Surie *et al.* (2007) uses a rapid trust sniffer module to enhance security. It involves moving trust partially from application layer to operating system layer. The trust sniffer validates an application by comparing sample measurement with reference measurements. Trust sniffer aims to enhance security with modest user effort. A key design principle, motivated by ISR's unique characteristics, is to validate only the software a user needs for a task.

Boukerche and Ren (2009) discussed the characteristics and security issues with wireless and pervasive data communications for a ubiquitous and mobile healthcare system which consists of a number of mobile devices and sensors. A secure multicast strategy is proposed that employs trust in order to evaluate the behavior of each node, so that only trustworthy nodes are allowed to participate in communications, while the misbehavior node is effectively prevented.

YuLong *et al.* (2010) assured availability of resources, trustworthiness by creating an explicit trust binding between the components that may participate in a service composition. A CTB is a set of rules which define the collection of allowable components for a particular service. A service-invoking node can distribute a CTB to the service providing node which is then expected to enforce the CTB policy as to which components are permissible for use in delivery of the service. Similarly a content owning node can distribute a CTB to a service which processes the content, which is then expected to enforce the CTB policies during access to that content.

Bonatti *et al.* (2010) described PROTUNE, a rule-based Trust Negotiation system and the advantages that arise from an advanced rule-based approach in terms of deployment efforts, user friendliness, communication efficiency and interoperability are illustrated.

Ahamed *et al.* (2010) has proposed an approach to addressing offer generation ,including the architecture, the information modeling and the generation algorithm. The roles of both parties are studied and focus on how

service providers generate offers upon receiving the requests from service requesters.

Buford *et al.* (2006) considered a problem of mediated group decision making where a number of agents provide a preference function over a set of alternatives. A mediation step is applied to aggregate the individual preferences in order to obtain a group-preference function and the most supported alternative is selected. It is proposed to define a scoring or preference function as the solution of a nonlinear optimization problem. The model also takes into account that imprecision could exist in the preference functions.

Manchala (1998) was one of the first members to quantify trust. He explained and established metrics such as transaction cost, transaction history, indemnity and various models to establish mutual trust across customer and venders. He also provided various trust models to analyze the risk involved in a transaction based on trust.

Timothy and Ramakrishnan (2003) presented the Joint control of Virtual Organizations (JoVO), a framework based on joint control of identity, attributes and access control in a virtual organization through the use of threshold based certification authorities They proposed an automated distributed audit agent framework consisting of white-box and black-box service testing for joint validation of access control policy.

Smith *et al.* (2004) and Seamons *et al.* (2002) outlined the requirements for policy languages for trust negotiation. The requirements are used to evaluate existing policy languages such as PORTFOLIO and service protection language, Trust Policy Language. With the goal of influencing policy Language designers they extend existing languages or designs new languages. X-sec is an XML based language for specifying credentials and security policies for web document protection.

Ye *et al.* (2004) presented a collaborative trust negotiation scheme, instead of reputation in P2P system. In their scheme, peers build trust with each other through the exchange of digital credentials. They introduce the concept of locally Trusted Third Parties to enable the collaborative trust negotiation and solve the problem of cyclic credential disclosures policy interdependencies. In addition, a credential issuer, which is a trusted third party for all peers called the Globally Trusted Third Party (GTTP) is also introduced.

Proposed architecture: The working of proposed architecture is discussed in Fig. 1. When a consumer requires a specific type of service, it sends its request to any of the associated community head that it may be interested in. In its request, the consumer specifies its constraints on the type of the requested service, the cost, the schedule and the required policy constraints.

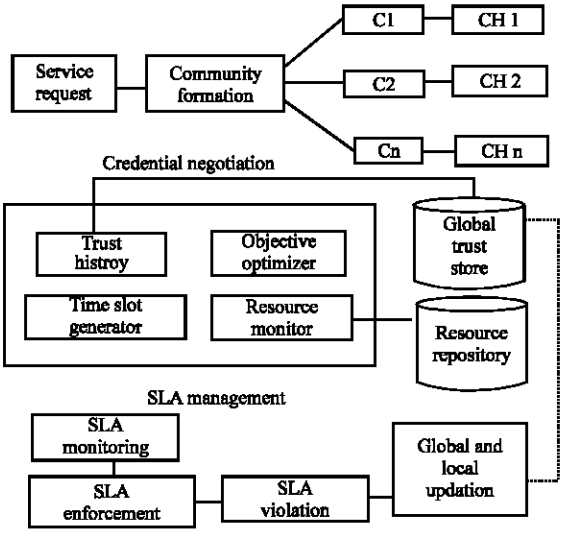

Fig. 1: SLA based trust model

The community head examines the trust-indices of all its associated agents who could provide the requested resource as per the consumer's constraint. To avoid traffic congestion, only the top T agents, whose trust-indices are greater than the threshold-index, are chosen for the delegation of the consumer's request. To avoid agent-starvation, the community head may also decide on random selection of T Agents from among the agents attached to it. Here, the community head will choose the best trustworthy P under multiple agents.

The consumer requests include certain policy requirements that the P should posses. Considering the RP's point of view, when a P associates itself with a agent, it specifies a set of policy requirements that the consumer should possess to utilize its resources. The agent processing the consumer request tries to find a P that not only matches the consumer's constraints with good trust-index for providing trustworthy resource but also the one with matched policy requirements of both consumer and P. The agent nominates an P who satisfies both the trust-index and policy requirements of the consumer's request to the associated Head.

```
    If (consumer's request constraints match P's constraints)
    If (consumer's policy-constraints match P's policy-constraints)
Agent nominates the P with the highest trust-index to the Head
Else
    Agent sends the failure notification to Community head
End if
Else
    Agent sends the failure notification to Community head
End if
```

In case of more than one provider complying with the selection criteria, the process of choosing a suitable P for a transaction involves a hierarchical priority-based selection procedure. This selection procedure consider (1) The consumer's constraints, (2) The trust-index of the Agents and the Provider in addition with the matched-policy constraints, (3) Queue length at the P and finally(4) The consolidated cost of past transactions of the Provider. Each of these T Agents selects Provider, having the highest trust-index among all the Provider that best satisfy the consumer's constraints with required policy-constraints specified in the consumer's request (Li *et al*., 2009). Also nominates those Providers to the associated community head as its choice, along with Provider's policy-constraints required from the consumer side.

The community head examines the suitability of all the Providers nominated to serve the request and selects the one with the best trust-index with the matched consumer's and Provider's policy-constraints. In the case of more than one Provider complying with the selection criteria, similar selection procedure is adapted. The community head intimates the selected Provider about the details of the consumer's request through the agent, who nominated this Provider. The consumer is also notified of the selection.

The consumer utilizes the services of the selected P through their associated Agent. After the completion of transaction, the feedback about the service is received from both the consumer concerned and the associated agent by the associated Community head. The Head will check the genuineness of the feedbacks generate the genuine feedback by itself in case of malicious feedbacks (Ahamed *et al*., 2010). Then update the trust-index of consumer, Agent and P based on their genuineness in the current transaction along with previous trust-index.

Request processing: Node A, when it wants to avail a particular service from the community, it sends a message in the format specified previously to the community head. On receiving the message, the community head now retrieves the required credential list from the message. From this list, it derives a priority or weight for every corresponding policy j, for every node i as w_{ij}.

Algorithm: Request processing

Input: Service request message
Output: Set N that contains nodes after successful credential matching
or every node i,
    assign priority w_{ij} to all the credentials;
1<=j<=n
P : = Nodes providing service S
For every node i in P,
    p_{ij} _credential value for node i under credential j;1<=j<=n
$X_i = \sum w_{ij} p_{ij} / ?w_{ij}$
If(X_i >=X)
    Then N=N U {i}
If N = Ψ
Forward request to neighbouring communities
End

For all nodes i in N.

$$\psi_i = X_i + \left(\alpha * \frac{N_s}{N_s + N_F}\right) + \beta * T_i \quad (1)$$

If, $\Psi_i < \text{Treshold}_A$, remove i from N, N_S, is the number of successful interactions, N_F, is the number of failed interactions.

$$\alpha + \beta = 1 \quad (2)$$

Equation 1 takes into account three components:

- The first component in the equation has major weight in the selection process. It takes into account the credential matching factor for a service provider selection
- The second component takes into account a part of the local trust, the requestor has on a provider node interms of successful integrations in previous history
- The third component takes into account the global trust of provider node. The global trust is pooled together by the agent, as requesters send their local trust values to the agent

Negotiation: Instead of going by the node by the maximum value of Ψi. Node A negotiates with the nodes in N to find out the best offer that is optimized in time and cost (Ismail *et al*., 2010).

while (!isEmpty(N))
    j_ retrievenode(N);
if(time(j)<=obt && cost(j)<=C)
    Z=Z U {j}
A= AU {Zi} where time(Zi) is minimum.
If |A|>=1

Find j in A, such that cost (A) is min.

The negotiator internally interacts with the decision model to negotiate with the provider to make the decision efficient. The decision model is composed of the following components.

Resource monitor: The Resource monitor keeps track of all the resources or services that can be provided by any node. At any given instance of time, the resource monitor

will help determine if a resource or service is available for use, number of resources available, no of resources free to use etc. It also provides the resource usage information during the phase when a provider is being negotiated.

Time slot generator: It allots specific time for using the individual resources based on their availability and certain other criteria such as requester's demand over the resources, resource availability.

Objective optimizer: It is a component that attempts to determine the best time slot and resources for a requested service. It is estimated based on the timeslot where the service when scheduled, yields a best fit result.

Trust evaluation and updation: After every interaction, the trust value a node has on another, has to be updated based on how well the provider responded. A direct trust computation is performed after two nodes have finished interacting with each other (Denko *et al*., 2010). Let us consider two devices A and B. $T_A(B)$ is used to represent the trust that node A has on node B. After every interaction between A and B, their trust is aggregated and stored locally by A and also contributes to global trust value of B. This value is calculated based on the outcome of the interaction, the satisfaction of node A with node B's service.

Basically node A performs an evaluation of the service provided by node B, based on the compliance of the agreement made beforehand and gives a rating value r. time, which is the time taken by the node to respond is also included in the evaluation process. The more the time, less satisfied node A would be:

$$T'_A(B)=\omega*\frac{T_A(B)^{\frac{r}{time}}}{1-T_A(B)} \quad (3)$$

where, $T'_A(B)$ is the updated trust value and ω is the quantization factor that is used to keep trust in the range 0-1.

To update the global trust value:

Step 1: Get tuple corresponding to node B <T(B),n,t>
Step 2: Calculate $T'_A(B)$:

$$T'_A(B)=\frac{n*T(B)+T'(B)}{n+1} \quad (4)$$

Step 3: Get current time, t
Step 4: Update tuple in global set as < T'(B),n+1,t>

The Algorithm makes use of weighted arithmetic mean to update the trust value globally by maintaining trust as an average of all the users in the environment. The global trust of a node or device builds up only gradually and there is no way of one particular entry, malicious or not, to malign the existing values.

SLA violation: Monitoring SLA Violation begins once an SLA has been defined. A copy of the SLA must be maintained by both the client and the provider. A request to invoke a service based on the SLOs (which are the SLA terms), for instance, may be undertaken at a time much later than when the SLOs were agreed. During provision it is necessary to determine whether the terms agreed in the SLA have been met (Chi *et al*., 2011). A monitoring infrastructure is used to identify the difference between the agreed upon SLO and the value that was actually delivered during provisioning. It is also necessary to define what constitutes a violation. Depending on the importance of the violated SLO and/or the consequences of the violation, the provider in breach may avoid dispatch or obtain a diminished monetary sanction from the client.

An SLA may be terminated in three situations: (1) when the service defined in the SLA has completed; (2) when the time period over which the SLA has been agreed upon has expired and (4) when the provider is no-longer available after an SLA has been agreed (for instance, the provider's business as gone into liquidation). In all three cases, it is necessary for the SLA to be removed from both the client and the provider. Monitoring can be used to detect whether an SLA has been violated. Typically such violations result in a complete failure-making SLA violations an 'all-or-nothing' process. In such an event a completely new SLA needs to be negotiated, possibly with another service provider, which requires additional effort on both the client and the service provider.

SLA monitoring: In order to enable SLA enforcement, an understanding of the current and recent state of the underlying resources is required. Umuhoza *et al*. (2007) discussed resource availability and utilization can be sampled periodically in a coarse-grained manner in order to provide a high-level understanding of general Quality of Service (QoS) indicators. At other times it may be appropriate to target particular and detailed attributes that reflect a given resources' ability to fulfill a particular action, e.g., the execution of a job. The rating r specified in Eq. 3 is obtained from the compliance and deviance values of the agreement. Here, we considered SLA parameters are:

- **Turn around time:**

$$r_1 = 1 - \frac{\Delta TAT}{ExpectedTAT} \quad (5)$$

- **Packet delay:** Two equations are used to calculate trip time variation (ΔTt). Trip time Tt is calculated using the following equation:

$$T_t = T_r - T_s \quad (6)$$

where, T_s is the time a packet is sent and the time it is received is T_r and the estimated reference time T_R is given by the equation below:

$$T_R = \frac{(RQ_r - RQ_s) + (RP_r - RP_s)}{2} \quad (7)$$

Where:
RQr = Time the route request message is received
RQs = Time route request message is sent
RPr = Time route reply message is received
RPs = Time route reply message is sent

Having the value calculated in Eq. 7, we can calculate the variation of trip time of each packet. The variation of trip time is calculated during route discovery process by the following expression:

$$\Delta T_t = T_r - T_s \quad (8)$$

The trip time of one packet alone is not meaningful but observing trip time variations over a period of time will allow the computation of probability of a packet to be delayed. Comparing trip time variation of many packets helps in noticing and examining regular delays that are most likely to be caused by attacks.

With the aid of Eq. 6 and 7, the probability that a packet is not delayed in the path (network) by an external influence is as follows:

$$P_u = \frac{T_t}{T_R + b\sigma} \quad (9)$$

where, b is a constant which we assume lies in the range $2 \leq b \leq 5$ and σ is standard deviation in trip time of the packet. The choice of b is empirical and may be evaluated by experimentation.

If the trip time of a packet is bigger than the denominator of Eq. 9 we assume that an unusual event has occurred delaying the packet from arriving on time and hence the path should be less trusted and changed. Therefore, based on the Eq. 9, the trust update probability is:

$$\Delta P_u = \frac{\Delta T_t}{T_R + b\sigma} \quad (10)$$

Therefore rating based on packet delay variations is P_{tt}:

$$r2 = P_{tt} + \Delta P_{tt} \quad (11)$$

- **Probability of packet loss:** Probability of packets lost is computed as:

$$= (\pi nl / \pi ns) \quad (12)$$

$$r3 = 1 - P_{loss} \quad (13)$$

- **Throughput:**

$$r4 = 1 \frac{\Delta Throughput}{Expected\ throughput} \quad (14)$$

- **Energy consumption:**

$$r5 = 1 \frac{\Delta Energy\ consumed}{Expected\ consumption} \quad (15)$$

SLA enforcement: The SLA Enforcement is tasked with monitoring SLA fulfillment. The SLA Enforcement uses monitoring data from the Resource manager, Timeslot Generator. When an SLA violation is detected, the SLA Enforcer takes reactive measures such as SLA re-negotiation or compensation retrieval based on SLA penalty clauses.

SIMULATION AND RESULT

Formation of community and registration: All nodes are sorted into communities. The communities are based geographically and are clustered together. Once communities are formed, all nodes are registered with their respective community heads. The community heads act as agents to find appropriate provider nodes for any requester. So, it is mandatory for any node to register with the community head with its necessary details. In Fig. 2 requester sends service request to its respective community head. The community head returns a list of providers to the requester. The requester on receiving the community head's recommendation, sends a request message to all the providers in the list.

All nodes that receive this request message respond to the sender by sending their earliest time by which the request could be processed and the cost involved for that process. In Fig. 3, the requester collects responses from

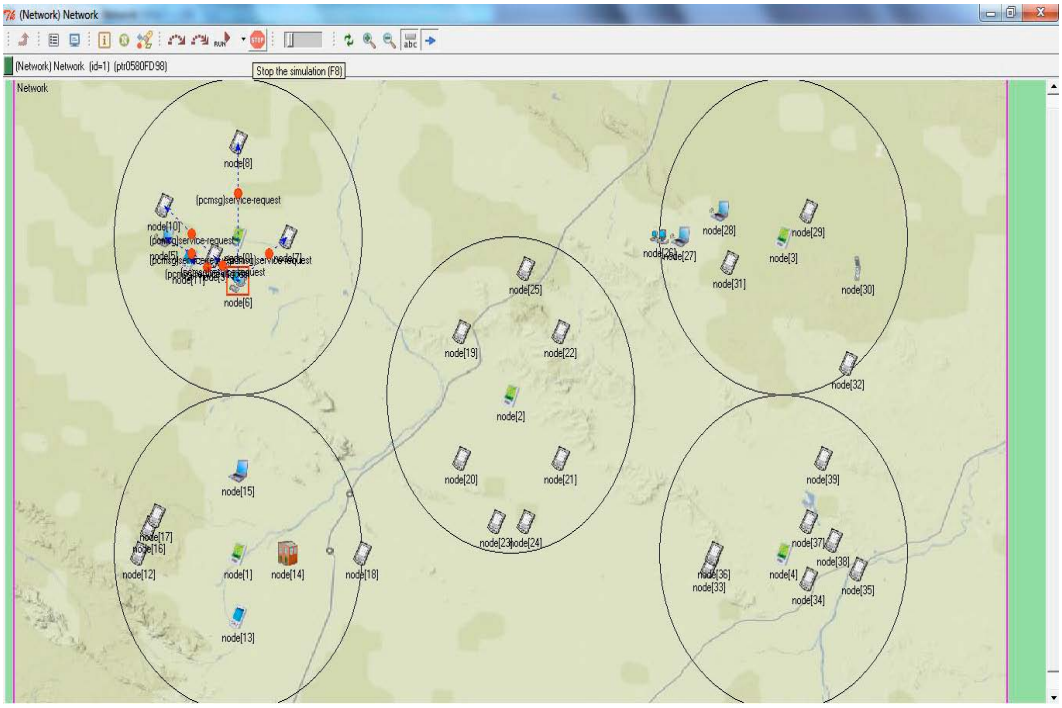

Fig. 2: Service requester send request to all providers for selecting the best provider

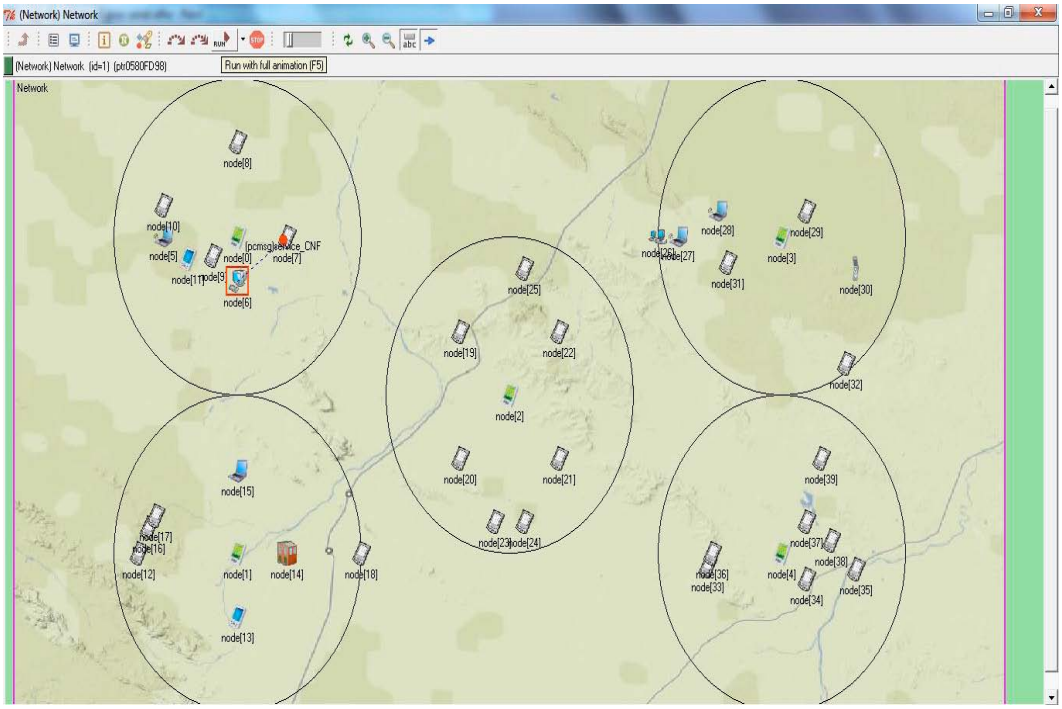

Fig. 3: Selecting Best Provider from list of providers obtained from community head

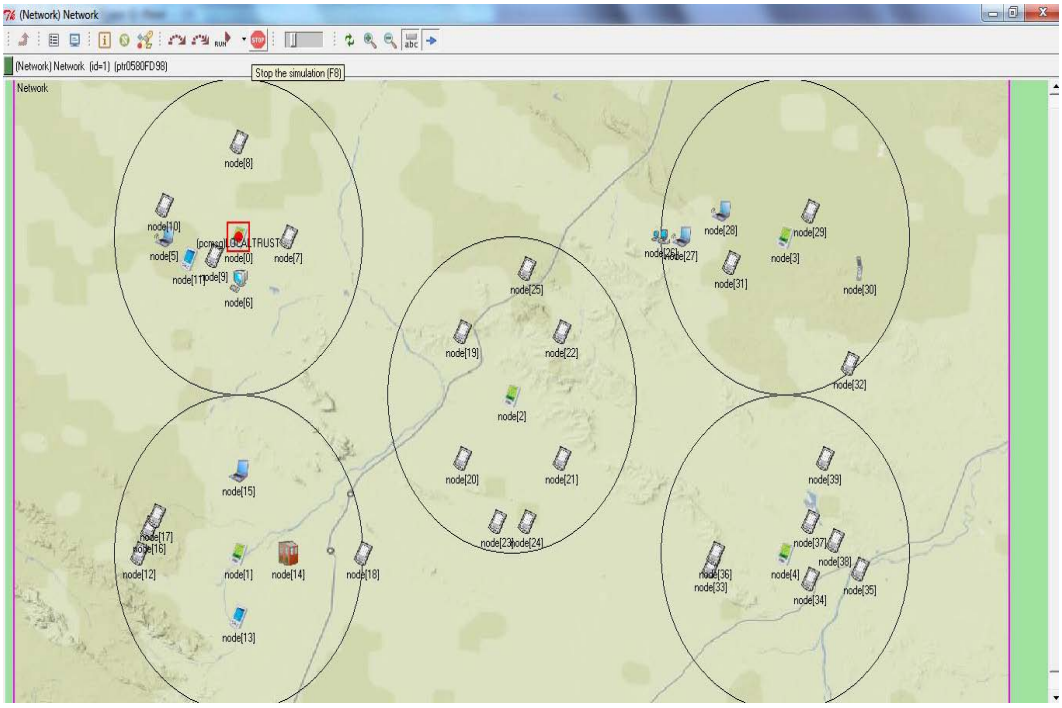

Fig. 4: Global updation of aggregated trust value

the nodes until certain time and processes all the offers to choose the best in terms of time and cost and trust index.

After a provider is chosen, the interaction between requester and the chosen provider takes place. The interaction monitored closely for Turnaround time, Packet drops, throughput and energy consumption rate and packet delay. The deviation in these parameters is used to calculate the trust value of the provider. The trust is first updated locally by the requester. Then it is sent to the community head. The community head aggregates the global trust (recent trust and past trust) for the provider and the updated trust value is shown in Fig. 4.

Cost-loss to consumers is plotted against the number of transactions in the following three scenarios-selection of Provider based on trust, based on trust and policy and the selection without trust. Cost-loss is defined as the fraction of cost incurred by the consumer for unsatisfactory service. Cost loss under various cases is plotted in Fig. 5. Trust index form the basis of selection of provider (Orwat *et al*., 2010). Cost-loss is more for the selection of provider with no trust index. And also it is more for the selection of P based on trust alone compared with trust and policy matched trust, which is shown in Fig. 5. This shows the importance of trust and policy matched trust-index for the selection of suitable provider and minimizing the cost-loss for

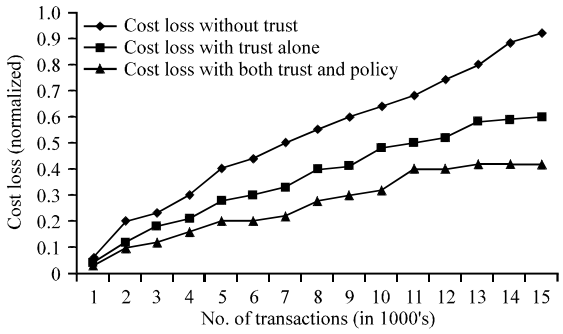

Fig. 5: Comparison of cost-loss with trust alone, without trust, trust with matched policy

the consumer community. This indicates the robustness of the use of trust-index in the selection process.

Job success rate is plotted in Fig. 6 against the percentage of the malicious Agents with different threshold values with trust. In this, the job success rate is much higher and stable compared to the earlier Architecture with increasing the percentage of malicious Agents. This is because the biased agents who provide biased feedback is considered as the satisfaction-index of the present transaction. Hence, the job success rate remains unaffected despite the increase in the percentage

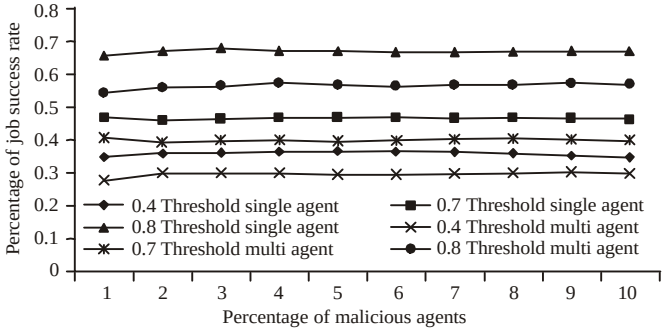

Fig. 6: Job success rate vs. malicious agents

of malicious agents. Also as the threshold value increases, job success rate increases. So higher the threshold, stricter is the selection process and higher is the job success rate.

CONCLUSION

In this study an SLA based trust management is proposed. Using Service Level Agreement, both the parties have an understanding of what is expected out of them. The customer knows exactly what to expect in terms of quality of service. This enables the consumer to receive the response from the Agent significantly quicker compared to other trust models where the trust-indices are computed at the request-time. Since the Agents have ample data to support the choice of a suitable Provider, the Agents, on behalf of the consumers, handle the selection efficiently. Thus, the consumer is relieved of the risk of selecting an unsuitable RP with limited information. Experimentally, it is observed that trustworthiness of node is maintained steadily with the number of interactions. It is also observed that the nodes with ill intensions are identified in 40 % lesser interactions than in the conventional models.

REFERENCES

Ahamed, S.I., M.M. Haque, Md. Endadul Hoque, F. Rahman and N. Talukder, 2010. Design, analysis and deployment of omnipresent Formal Trust Model (FTM) with trust bootstrapping for pervasive environments. J. Syst. Software, 83: 253-270.

Bonatti, P., J.L. De Coi, D. Olmedilla and L. Sauro, 2010. A rule-based trust negotiation system. IEEE Trans. Knowledge Data Eng., 22: 1507-1520.

Boukerche, A. and Y. Ren, 2009. A secure mobile healthcare system using trust-based multicast scheme. IEEE J. Selected Areas Communi., 27: 387-399.

Buford, J., R. Kumar and G. Perkins, 2006. Composition trust bindings in pervasive computing service composition. Proceedings of the 4th Annual International Conference and Pervasive Computing and Communications Workshops, March 13-17, 2006, Pisa, pp: 261-266.

Bussard, L., Y. Roudier and R. Molva, 2004. Untraceable secret credentials: Trust establishment with privacy. Proceedings of the 2nd Annual Conference on Pervasive Computing and Communication Workshop, March 14-17, 2004, Orlando, Florida, pp: 122-126.

Chi, Y., H.J. Moon and H. Hacigumus, 2011. iCBS: Incremental costbased scheduling under piecewise linear SLA. VLDB Endowment, 4: 563-574.

Denko, M.K., T. Sun and I. Woungang, 2010. Trust management inubiquitous computing: A Bayesian approach. Comput. Communi., 34: 398-406.

Ismail, A., J. Yan and J. Shen, 2010. An offer generation approach to SLA negotiation support in service oriented computing. Service Oriented Comput. Applic., 4: 277-289.

Li, L., M. Song, X. Zhang, 2009. An SLA based web service quality monitor system. Proceedings of the Joint Conferences on Pervasive Computing, December 3-5, 2009, Tamsui, Taipei, pp: 661-664.

Manchala, D.W., 1998. Trust metrics, models and protocols for electronic commerce transactions. Proceedings of 18th International Conference on Distributed Computing Systems, MAy 26-29, 1998, Amsterdam, pp: 312-321.

Orwat, C., A. Rashid, C. Holtmann, M. Wolk, M. Scheermesser and H. Kosow, A. Graefe, 2010. Adopting pervasive computing for routine use in healthcare. IEEE Pervasive Comput., 9: 64-71.

Pelta, D.A. and R.R. Yager, 2010. Decision strategies in mediated multiagent negotiations: An optimization approach. Syst. Man Cybernetics, Part A: Syst. Humans, IEEE Trans., 40: 635-640.

Seamons, K.E., M. Winslett, T. Yu, B. Smith and E. Child *et al*., 2002. Requirements for policy languages for trust negotiation. Proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks, June 5-7, 2002, Monterey, CA., pp: 68-79.

Shuai, Z., X. Yang, Y. Yi-Xian, H. Zheng-Ming, 2010. TM-RMRP: A trust model based on a recommendation mechanism of rewards and punishments. Proceedings of the 2nd International Workshop on Education Technology and Computer Science, Volume 3, March 6-7, 2010, Wuhan, pp: 31-34.

Smith, B., K.E. Seamons and M.D. Jones, 2004. Responding to policies at runtime in trust builder. Proceedings of the 5th International Workshop on Policies for Distributed system and Networks, June 7-9, 2004, York town Heights, pp: 149-158.

Sun, T. and M.K. Denko, 2008. Performance evaluation of trust management in pervasive computing. Proceedings of the 22nd International Conference on Advanced Information Networking and Applications, March 25-28, 2008, GinoWan, Okinawa, Japan. pp: 386-394.

Surie, A., A. Perrig and D.J. Farber, 2007. Rapid trust establishment for pervasive personal computing. Pervasive Comput., 6: 24-30.

Timothy, J.S. and L. Ramakrishnan, 2003. Joint policy management and auditing in virtual organizations. Proceedings of the 4th International Workshop on Grid Computing, November 17, 2003, Phoenix, AZ, USA., pp: 117-125.

Traek, D., 2011. Trust management in the pervasive computing era. IEEE Security Privacy, 9: 52-55.

Umuhoza, D., J.I. Agbinya and C.W. Omlin, 2007. Estimation of trust metrics for MANET using QoS parameter and source routing algorithms. Procedings of the 2nd International Conference on Wireless Broadband and Ultra Wideband Communications, August 27-30, 2007, IEEE., Pages: 80.

Ye, S., F. Makedon and J. Ford, 2004. Collaborative automated trust negotiation in peer-to-peer systems. Proceedings of the 4th International Conference on Peer-to-Peer Computing, August 22-27, 2004, Zurich, Switzerland, pp: 108-115.

YuLong, L., G. ShiZe, T. Ye and D. YouZhi, 2010. Trust propagation model in pervasive computing environment. Proceedings of the 1st International Conference on Pervasive Computing Signal Processing and Applications, September 17-19, 2010, Harbin, pp: 120-123.