Security Framework for Agent-Based Cloud Computing

Venkateshwaran K, Anu Malviya, Utkarsha Dikshit, S.Venkatesan

Department of Information Technology, Indian Institute of Information Technology Allahabad

***Abstract* — Agent can play a key role in bringing suitable cloud services to the customer based on their requirements. In agent based cloud computing, agent does negotiation, coordination, cooperation and collaboration on behalf of the customer to make the decisions in efficient manner. However the agent based cloud computing have some security issues like (a.) addition of malicious agent in the cloud environment which could demolish the process by attacking other agents, (b.) denial of service by creating flooding attacks on other involved agents. (c.) Some of the exceptions in the agent interaction protocol such as *Not-Understood* and *Cancel_Meta* protocol can be misused and may lead to terminating the connection of all the other agents participating in the negotiating services. Also, this paper proposes algorithms to solve these issues to ensure that there will be no intervention of any malicious activities during the agent interaction.**

***Keywords* — Agents, Cloud Computing, Security, Contract Net Protocol, Service Capability Table, Agent Trust Table**

I. INTRODUCTION

CLOUD computing is a fast developing technology which provides scalable data storage to large and various services without the hassle of installation and maintenance. Since there is an increase in number of users for the cloud services, there is a demand on cloud service providers. Hence there is a need for dynamic and automated cloud service composition [1, 2].

With the emergence of large number of service providers, users are not able to choose the best cloud service based on their technical and financial requirements.

To address this problem agents are introduced in the cloud environment. These agents make the decision making process easier for consumers by choosing and providing the best fit service for them, based on their requirements. According to Kwang Sim's model of agent based cloud composition [1], there are four agents involved in the cloud commerce such as Consumer Agent (CA), Broker Agent (BA), Service Provider Agent (SPA) and Resource Agent (RA).

Every agent will maintain a SCT (Service Capability Table), the attributes of SCTs are: (i) agents' addresses (ii) the requirements that agents can resolve, and (iii) the last known status of the service [1]. The SCT gets updated with agent's status after each and every agent-agent interaction.

All the agents interact with each other to deliver the best cloud service to the user. The whole process of agent interaction is controlled by the semi recursive contract net protocol (SR-CNP). This protocol is used to do the negotiation process between the task managers and the contractors. Here, the agent who initiates the process and requires a task to be done is referred as task manager and the agent who is able to execute the task is known as contractor.

There are various interaction protocols that can be followed for the agent-agent interaction. The model of agent based cloud commerce requires recursive call for, proposal and acceptance at various stages. So the contract net protocol in a semi recursive manner suits well into the model.

There are two roles in the contract net protocol: (i) Initiator (ii) Participant [6].

A consumer adopting the initiator role broadcasts a call-for-proposals to achieve a task (e.g., service composition) to n participants (contractors). The participants may reply with: (i) a proposal (quotation) to carry out the task, or (ii) a refuse message.

From the received m proposals, the initiator will select the best (cheapest) proposal, and sends: (i) an accept-proposal message to the best participant, and (ii) reject-proposal messages to the remaining (m – 1) contractors [1].

After carrying out the task, the selected participant sends either: (i) an inform-result message or (ii) a failure message in case of unsuccessful results.

This paper briefs about the mechanism of agent based cloud computing in section II, explains the security issues with agent based cloud computing in section III, and proposes solutions to overcome the issues in section IV and conclusion in Section V.

II. MECHANISM OF AGENT BASED CLOUD COMPUTING

A. Consumer Agent

Consumer agents receive requirements from the consumers. Each consumer agent maintains an SCT, which contains list of several broker agents known to it. Whenever a user requests for a cloud service, consumer agents receive these requests and sends a call for proposal to all the broker agents with a certain timeout, say 30 seconds, to respond.

Broker agent responds with either accept or reject based on its ability to resolve the request. Consumer agent only accepts those responses which come within the timeout period, other responses are discarded.

Among all the responses received, CA selects the most suitable BA, sends the accept-proposal to the selected BA and

DOI: 10.9781/ijimai.2015.334

refuse-proposal to all other BAs.

B. Broker Agent

Broker agents provide a single virtual cloud service to the consumers by contacting and selecting set of Service Provider Agents (SPA). BAs act as an intermediate between the CAs and SPAs.
Every BA has two SCTs:

a) List of SPAs

b) List of other BAs (to be used in case of sub contracts required).

BAs also handle the update requests from the consumer agent. BA selected by the consumer agent selects the SPAs from its list, makes a contract with SPA and delivers service to the consumer agent.

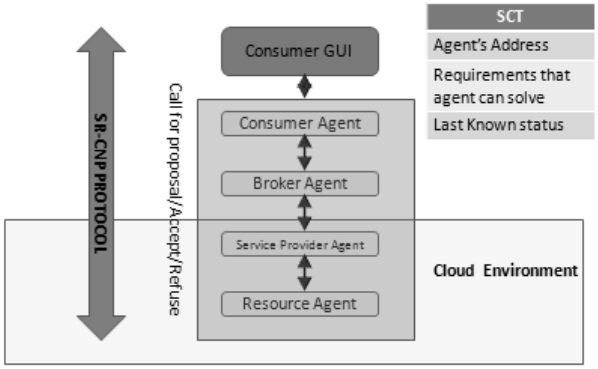

Fig 1. Mechanism of Agent-Based Cloud Computing

C. Service Provider Agent

On the agreement of transactions, SPA allocates and de-allocates the cloud resources from the resource agents.
Every SPA has two SCTs.

a) List of Resource agents (RA)

b) List of other SPAs for the subcontracts.

SPAs keep track of the available resources and synchronize with the RAs for concurrent or parallel executions. Selected SPA approaches the available RAs and makes the contract for the consumer requirements.

D. Resource Agent

Resource agents are the major control agents for accessing cloud resources. RAs are associated with SCT table consisting of SPAs. Whenever there is a request from SPA, RA sends resource or status to SPA based on the availability of resources.

As depicted in Figure 1,Once RA sends resource to SPA, resources are delivered to BA, BA delivers the cloud service to CA and consumer gets its service from CA.

Agents use predefined built-in functions [1] for sending messages to other agents.

The process is bounded by the timeouts. It involves two timeouts, timeout1 and timeout2. Timeout1 refers to the deadline of proposal submission and timeout2 refers to the deadline to deliver the virtual service.

This mechanism, delivers the cloud service to the consumer by making use of agents.

III. SECURITY ISSUES IN AGENT BASED CLOUD COMPUTING

Agent based cloud computing is developed in an ideal environment. Agents have been introduced to mainly focus on the process of negotiation for choosing the best cloud resource for the consumer. Since agents are the third parties, there are lots of security issues involved. This paper identifies several security issues which can block the agents from choosing suitable resources.

1. Addition of Malicious agent

Unlike the acquaintance network which updates the agent list only during the addition of new agent, SCTs update the agent list whenever a transaction happens between the agents. Though this feature of SCTs improves the performance of message exchange and always keeps the updated information about agent in the table, there is a security threat in addition of new agents.

According to Kwang's model [1], SCTs can add a new agent into the list when there has been a previous encounter with the agent or by mere presence of an agent in the same cloud.

In this scenario, any malicious agent can add itself into an SCT and can receive all the consumer requirements associated with it.

Following are the possible impacts when a malicious agent gets added into an SCT.

a) Getting involved in all consumer requirement negotiations thus misguiding the process by providing unrealistically cheap prices and blocking other legitimate cloud resources from providing services to the consumer.

b) Capturing the responses of other agents and sending spoofed messages to the initiator and other participant agents.

2. Flooding Attack

To keep the records updated, SCTs get updated with agent's status whenever a transaction between agents occurs.For Example consider, a broker agent sends a call for proposal request to all the SPAs given in the SCT. Suppose the broker agent's SCT contains a malicious agent then during the broadcast of call for proposal for a consumer requirement, malicious agent gets a message. Now, the malicious agent can flood the response to the initiator agent (i.e. Broker agent) with its response as accept the proposal and status of the agent as available.

Until the timeout, initiator agent will receive all the responses from SPAs and update its SCT. When a malicious agent creates flooding response, SCT will be involved in updating the information of the malicious agent only.

3. Exceptions to Protocol flow

FIPA has mandated few exceptions in the agent interaction which should be present in every multi-agent system to control the flow of the process. Some of such exceptions like Cancel_Meta protocol and Not-Understood problem can be misused by the intruder agents.

These exceptions can be used for attacks as explained below:

(i) Forced termination of agent interaction

As per FIPA interaction protocol flow [6], any interaction between agents is identified using a globally unique and non-null conversation-id parameter. In a multi-agent environment having no security measure, a malicious agent can get involved in some other agent-agent interaction.

A broker agent sends a call for proposal to all the SPAs in its SCT. Agents reply with accept/reject messages. Any malicious agent can send a spoofed message with conversation-id and agent's address stating that context of the message is not-understood.

Not-Understood is a communicative act in the FIPA so that an agent should be able to handle errors when the semantics followed by different agents are different. When any agent does not understand the context of message sent by the sender, then the receiver can send a Not-Understood message, in this case sender will handle the error and terminate the connection with receiver. This can be exploited by malicious agent because on receiving the Not-Understood message, Broker agent terminates the connection with the SPA.

Further, response of the legitimate SPA will be discarded by the broker agent. Hence, there will be a forced termination of the connection between agents.

(ii) Artificial timeout creation

When an agent sends any request to other agent, it receives the response within the timeout period. As per FIPA exception of protocol flow, there is a provision that a sender can cancel the previously sent request by sending a Cancel_Meta protocol message to the receiver. On receiving Cancel_Meta protocol message, receiver thinks that sender no longer requires response for the request sent.

In cloud environment, any malicious agent with conversation-id and agent address can send a Cancel_Meta protocol. On receiving the message, receiver ignores the request sent from the sender while sender is still waiting for the response from receiver until the timeout period. Hence the artificial timeout created by malicious agent stops the receiver agent from sending the response to sender agent.

IV. PROPOSED SOLUTION

Proposed framework consists of various modules includes Security Agent, serving as front end authenticator and trust analyzer of a cloud. Other sections depict solution for various identified security issues.

Security Agent

Among the four agents (CA, BA, SPA, RA) involved in cloud computing, SPA and RA are created by the respective clouds and are called as cloud agents. Remaining CA and BA are referred as outsider agents. These outsider agents especially BAs interact with SPAs to get a cloud's service.

Hence, entry of malicious agent may occurs when BA come to interact with an SPA to request and negotiate for a requirement. So, a new entity known as Security Agent (SA) is introduced for every cloud environment (Fig. 2) to handle the outsider agents.

SA provides two services:

i. Verification

ii. Trust Degree Analysis

To interact with SPA and RA of a cloud environment, an outsider agent should be authenticated by the Security Agent every time (Fig.1).

(i) Verification

When an agent comes to interact with any cloud environment, SA should verify the agent with Agent Trust Table. If the agent record is not available in ATT, it is considered as New Agent to the cloud environment. The agent details will be added to ATT after verification process from Third Party. If the agent record is available in ATT, agent is already registered by SA and considered as Registered Agent.

For Registered Agent, SA should check for agent's authentication on its proxy server with the credentials. If the authentication process fails the agent is discarded with no more further processing. If the authentication is successful, the agent is allowed to interact with the cloud agent and then the trust degree of the replying agent will be analyzed and updated in ATT.

Thus a secure environment for agent interaction can be created and this can resolve the addition of malicious agents into the cloud environment (Fig. 2).

(ii) Trust Degree Analysis

To maintain trust in agent interaction, a trust model can be used. When a trusted communication happens, the trust degree of the agent gets increase.

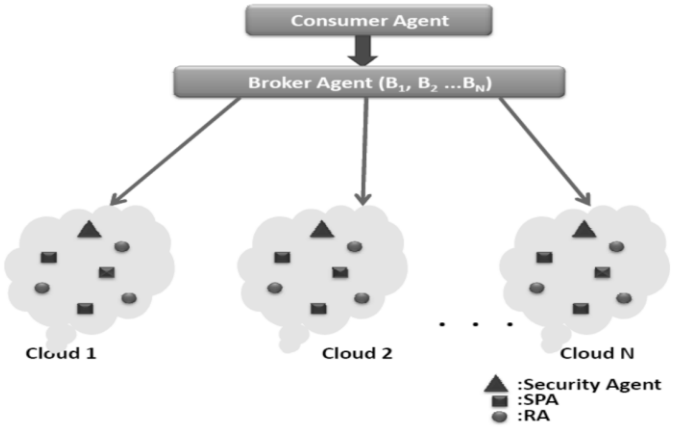

Fig 2. Agent Based Cloud Computing

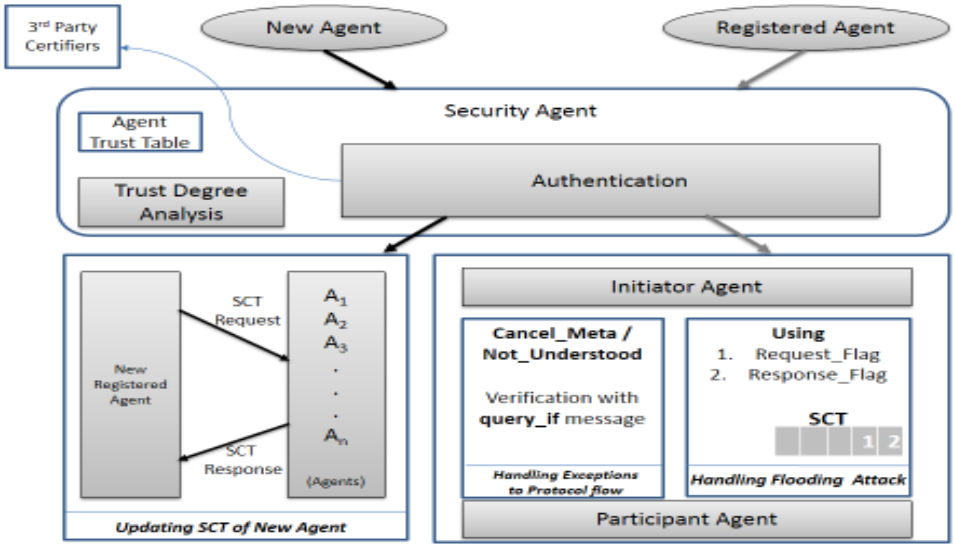

Fig 3. Framework for Secure Agent Communication

Similarly, when a non-trusted communication happens, the trust degree of the agent gets decrease. The probability of executing a request for any trusted agent is higher than the non-trusted or innocent agent. Suppose, there are n number of agents in a cloud with their Agent ID's = {AID1, AID2… AIDn}. If at any instance ith reply is analyzed for addition of its details in SCT table of New Agent (NA), then i Є {1, 2... n}. The Agent may be trusted, non-trusted or innocent.

Agent Trust Table (ATT)

A trusted agent's Trust Degree increases and decreases on completion of a process either successfully or unsuccessfully depending on its performance or set policies. Probability function is used to determine the trust degree of an agent replying with its SCT table. Based on the TD, agents will be marked as trusted, non-trusted or innocent. Actions for any task can be positive or negative.

There is a difference among the negative actions. It can be a wrong action or a malicious action. Positive actions are the right actions done by the trusted agent. Wrong actions are the bad actions that do not cause any damage or may cause damages done by the innocent agent and malicious. actions are harmful actions such as attacks done by the non-trusted agent.

The Trust Degree can be calculated by the equation: [4]

$$\textit{Trust Degree} \quad TD = \left(1 - \frac{N_a}{T_a}\right) A_w^{(s)} \quad \textit{Where } 0 \leq TD \leq 1$$

N_a = No. of negative actions
T_a = Total no of actions
A_w =weight of an action = 1 (for positive action)
0.9 (for negative action)
s = security level, s $\geq$ 1
Initially TD = 1; s =1
Threshold value =0.1

As the trust degree is calculated by exponential times of security level, if the positive action is happen with number of times (security level s =1,2…n), the term $A_w^{(s)}$ and should maintain the trust degree value . Hence, for positive actions, Aw is set to 1 and for negative actions, Aw is set to 0.9 to decrease the trust degree.

For example, suppose for a particular agent, Na is 50, Ta is 100 and the last updated behavior is positive and the s is 10th.

Then the Trust Degree from the above equation comes as TD = 0.5, which is greater than the threshold value i.e. the action is positive. Hence, its details will be added in the SCT.

Security Agent creates and maintains an Agent Trust Table (ATT) that include Agent's ID (AID), number of negative actions (Na) , total number of actions (Ta), Agents Behavior, security level (s) and action value or Trust Degree (TD) (Fig. 4).

Agent Trust Table				
Agent's ID (AID)	Number of negative actions (N_a)	Total number of actions (T_a)	Agents Behaviour	Security level (s)

Fig 4. Agent Trust Table.

It is used to check trustworthiness of either a newly created or previously registered agent through this calculated Trust Degree. The Agent Behavior of ATT is used to account the action weight of that agent depending upon its behavior either positive or negative. This ATT is updated every time after completion of a transaction.

On every completion of a transaction, this Trust Degree is calculated and ATT is updated with latest Agent Behavior by the Security Agent.

Addition of New Agent

Security Agent (SA) authenticates new agent arriving in cloud and handles Trust Degree for updating Service Capability Table (SCT) of that new agent.

When a New_Agent (NA) arrives in cloud environment, it goes to Security Agent (SA) which authenticates this NA by checking its Agent_id (AID) in trusted third part, if present,

Algorithm for Addition of New Agent:

Input: New Agent
Output: Addition or Discarding of New Agent

1 New_Agent (NA) arrives in cloud environment
2 NA goes to Security Agent (SA)
3 if SA (AID (NA) present in index of AMS)
4 Assign password for AID
5 Check Authentication on proxy server created by SA
6 if (AID (NA) && pwd == Correct)
7 Create SCT table
8 Broadcast SCT_details (AID)
9 if SCT_details (AID,reply (AID_i))
10 Send Request (AID_i ,TD) to SA
11 SA→ if (TD (AID_i) > threshold_val ?)
12 Update (ATT)
13 Update New_Agent (AID_i ,SCT)
14 Check for more Agent's reply Goto Step 9
15 else decrease (TD(AID_i))
16 Report AID_i action as negative to SA
17 if (no. of negative behaviour >= x)
18 Report AID_i as malicious to SA and Discard AID_i
19 else Goto Step 9
20 else Goto Step 8
21 else Discard NA

SA will assign a unique password. Here after authenticated by the cloud's proxy server created by Security Agent and SCT table of NA is created, otherwise this NA is discarded.

The NA now sends a broadcast message to all the other agents in the cloud environment to enter details into its SCT table. Since the issue was to avoid addition of malicious agent details, so the trustworthiness of the agent is measured for every ith reply coming with its SCT details. The NA requests SA to check the Trust Degree (TD) of ith agent, if it's greater than or equal to defined threshold value, the SCT detail of ith agent is updated in NA's SCT. If number of negative behavior identified by SA is greater than the threshold value, the reply is discarded.

This involves two processes:

(i) Authentication and
(ii) Trust Examining.

i. Authentication

Addition of new agent to the cloud environment: According to various research papers addition of agent can be based on

1. Trust: Where Certificate Authority (CA) serves as the root of trust.

CA issues these certificates only to those Principals who are trusted by the CA based on their harmless intentions and actions (Principal is a person who signs on behalf of the Agent code and is responsible for the behaviour of agent. Principal should be well aware of the workflow, behaviour and operational consequences of the agent).

2. Validation: When the owner registers the agent to the agent platform, this platform should validate the owner and log the request's source address.

Thus an agent arriving newly in a cloud environment must be signed for trust or be registered with a Third party who can guarantor for the Agent's behavior. This generates a unique identifier for each agent named as Agent Identifier (AID).

When a new agent (NA) wants to enter into a cloud, it reaches to Security Agent of that cloud which checks for its registration with Third Party by looking for its AID into their index, its Access permissions and its previous transaction or registration details with other clouds, to verify whether the coming agent is a legitimate agent or not. After verification if NA is found legitimate, SA assigns a password to it. The agent is now every time authenticated on cloud environment by its proxy server with this AID and password. Any discrimination from above checks leads to discarding of agent from interacting with cloud agents. The agent record is added in the Agent Trust Table (ATT) with default values. All the trusted agents of the cloud are added in the agent SCT broadcast list.

Algorithm for Solution of Forced Termination of Connection and Artificial Timeout:

Input: Reply from Particapant_Agent
Output: Accept or Discard the Reply

```
1  Initiator_Agent sends Call_for_Proposal
2  if Reply (Participant_Agent, Call_for_Proposal) == Accept
3       Connection (i) Initialisation
4       if Reply_i ( ) ==Cancel_Meta || Not_Understood
5        query_if (Reply_i, Reciever_Agent→Sender_Agent)
6          if query_if (Ack_i) ==True
7            Process Reply ( Cancel_Meta or Not_Understood)
8          else
          Ignore (Reply_i )
9  else
   Ignore (Reply ())
```

ii. Trust Examining

When a reply is received from an agent with its current SCT details, a request is sent to Security Agent (SA) with Agent_ID (AID) where current or updated Trust Degree (TD), present in Agent Trust Table (ATT) is checked or calculated and compared with Threshold Value. If the Agent's TD on an instance i is greater than the set Threshold Value, the ATT is updated and NA's SCT is updated with SCT details of replying agent (AIDi).

If current or updated TD is less than the threshold value, the action is said to be a negative action. TD of that agent (AIDi) is decreased as per the set policies and action is reported as negative or wrong to SA. If negative action occurs greater than or equal to x times, the action is reported as malicious and hence this replying participant agent (AIDi) is discarded from further processing.

The advantage is that the New_Agent (NA) remains unaffected when an identified participant agent does any malicious actions in the cloud environment. The Trust Degree of participant agent decreases accordingly with the malicious activities and the updating policies.

Handling flooding attack

To handle flooding attack issue, two flag attributes: *Request_Flag* and *Response_Flag* (*Fig. 5*), are introduced into the SCT along with the Agent's Address, Requirement provides and the Last Known Status. Since these are flags, so there values are either 0 or 1. Initially, both *Request_Flag* and

Response_Flag are 0. The values of these flags changes when a request is sent or a response is received for any agent.

Service Capability Table				
Agent's Address	Requirement provides	Last Known Status	Request_Flag	Response_Flag

Fig. 5. Service Capability Table

When an agent sends a message to other agent, represented as Request initiation, the Request_Flag in SCT sets to 1 and starts waiting for the response. As soon as the reply is recieved, it checks for the current status of flags i.e. if any response is received until now for that agent or not. If Request_Flag is equal to 1 and and Response_Flag is 0, the SCT of the Initiator_Agent is updated with Response_Flag as 1. Otherwise the response is discarded, showing that the reply from the agent for that request is already received. As soon as timeout occurs, the flag values are again set to 0.

Handling exceptions to Protocol flow

To avoid attacks on exceptions to Protocol flow the use of query_if function [5,10] is proposed. When Initiator_Agent broadcasts a Call_for_Proposal to all the other agents in the cloud, all the other agents reply either with accept or reject message depending on their willingness to communicate. The Initiator_Agent initializes the connection with all the agents replying as *Accept,* with unique Conversation_ID and *Reject* reply is ignored. If during the communication a *Not_Understood* or *Cancel_Meta* message is encountered, *query_if* function is initiated. The Receiver_Agent of these message sends a *query_if* message to the Sender_Agent and waits for the acknowledgment. If acknowledgment comes as *true*, the connection is terminated follow the message and act accordingly. Otherwise reply is ignored and the communication is continued.

Limitations and Implications

Though, we have mentioned that SA should refer to the trusted third party to verify the genuineness of an agent, it depends upon the cloud service providers to decide which trusted third party they want to believe. SA has to process each and every agent interaction occurs in a cloud. SA must be developed with the capability to handle maximum number of agents' queries at same time.

There may be possibility of discarding of an agent request, if the SA is not developed to handle multiple requests. However, this framework can be extended to determine the quality of the service offered by an agent. When the quality of the service can be compared, the user will get the most suitable service than the negotiation based on cost and time.

Algorithm for Flooding Attack

Input: Status Update Request from Participant_Agent
Output: Updating SCT or Discard the Update Request
SCT (Agent's Address, Requirement provides, Last Known Status, Request_Flag, Response_Flag)

```
Initially,
   Request_Flag = 0;
   Response_Flag = 0;
1   Initiator_Agent initiates Request ();
        Set Request_Flag = 1
2   receive Reply (Initiator_Agent, Participant_Agent)
3   if (Request_Flag == 1 && Response_Flag == 0) ==True
4        Update SCT ( , , ,1,1)
5   else
                  Discard Reply (Initiator_Agent, Participant_Agent)
6   if (Timeout)
7        Reset Request_Flag=0
           & Response_Flag=0
8   else
   Goto Step 2
```

V. CONCLUSION

Cloud computing is one of the futuristic technologies on which technology giants are counting. In future, number of users using the cloud computing is expected to increase gradually as there is a demand for cloud service exists. In such a scenario, there will no doubt that agents will play key role in selecting suitable services to users.

Since, it will be in the hands of agents to deliver a service to end user, agents should be free from attacks and bias. In this paper we have identified several security issues during the agent interaction. We have proposed solutions to handle those security issues. End User who uses the cloud services doesn't have any idea about how the agents are interacting and the service delivered is best among others or not. There is a possibility that malicious agent can involve in the process and

deliver wrong or malicious service to the user. So, we have used the trust degree analysis to decide whether the agent involved in the negotiation process is trusted or not. Analysis of this will help the proposed framework of security agent to allow only the trusted agents to deliver the service to end-user. However, several issues may arise when the agents plays dominant role such as determining the quality of the cloud service. With the proper security measures implemented in the cloud environment, agent based cloud computing will play as a platform for the consumers to use the perfect service.

REFERENCES

[1] Kwang Mong Sim, "Agent-based cloud service composition," Springer Science + Business Media, LLC2012

[2] Kwang Mong Sim, "Agent-based cloud commerce," Department of Information and Communications, Gwangju Institute of Science & Technology, South Korea

[3] Page, J.; Zaslavsky, A.; Indrawan, M., "Countering agent security vulnerabilities using an extended SENSE schema," Intelligent Agent Technology, 2004. (IAT 2004). Proceedings. IEEE/WIC/ACM International Conference on , vol., no., pp.183,189, 20-24 Sept. 2004
doi: 10.1109/IAT.2004.1342942

[4] Shantanu Pal, Sunirmal Khatua, Corresponding Author Nabendu Chaki, Sugata Sanyal, "A New Trusted and Collaborative Agent Based approach for Ensuring Cloud Security," in arxiv.org 2011
92 A. P. C. Road, University of Calcutta, India

[5] Fabio Bellifemine, Giovanni Caire,Dominic Greenwood in book "Developing Multi-Agent Systems with JADE,"
Telecom Italia, Italy.

[6] "FIPA Contract Net Interaction Protocol Specification,"
in http://www.fipa.org/specs/fipa00029/SC00029H.html

[7] Fatma Masmoudi, Monia Loulou,Ahmed Hadj Kacem, "Formal Security Framework For Agent Based Cloud Systems," in 2014 International Workshop on Advanced Information Systems for Enterprises ,ReDCAD Laboratory, University of Sfax, Tunisia.

[8] Suleiman Onimisi Aliyu and Kwang Mong Sim, IEEE Senior Member, "Minimizing Message Exchanges in Agent Based Cloud Service Composition," in Proceedings of the International MultiConference of Engineers and Computer Scientists 2014 Vol I,
IMECS 2014, March 12 - 14, 2014, Hong Kong

[9] Khushbu Virani, Dhara Virani, "Service Composition Based on Multi Agent in Cloud Environment," in International Journal of Engineering Research & Technology (IJERT) Vol. 1 Issue 9, November- 2012

[10] "FIPA Request Interaction Protocol Specification,"
http://www.fipa.org/specs/fipa00026/SC00026H.htm

[11] Juan Pablo Paz Grau, Andrés Castillo Sanz, Rubén González Crespo "An Evaluation of Integration Technologies to Expose Agent Actions as Web Services," in Practical Applications of Intelligent Systems 259-270, January 1,2014, Springer Berlin Heidelberg

[12] Gutierrez, C., "An Analysis Architecture for Communications in Multi-agent Systems," International Journal of Interactive Multimedia and Artificial Intelligence (IJIMAI), Special Issue on Artificial Intelligence and Social Application Vol 2 Number 1 Pagination 65-72, March, 2013

[13] Jordán Pascual Espada, Vicente García Díaz, Rubén González Crespo, Oscar Sanjuán Martínez, B Cristina Pelayo G-Bustelo, Juan Manuel Cueva Lovelle, "Using extended web technologies to develop Bluetooth multi-platform mobile applications for interact with smart things," Information Fusion Vol 21 Pages 30-41,January 31,2015, Elsevier

S.Venkatesan received his Ph.D in Computer Science and Engineering from Anna University,Chennai, Tamil Nadu, India. He is an Assistant Professor in Department of Information Technology at Indian Institute of Information Technology, Allahabad. He has published various papers in numerous international conferences and journals. He is an active member in Association for Computing Machinery (ACM) Cryptology Research Society of India (CRSI). His current research includes Mobile Agent Security, Cryptography and Cloud Computing and Social Network Privacy.

Anu Malviya is a MS (Cyber Law and Information Security) research student in Indian Institute of Information Technology, Allahabad. She completed her Bachelor of Technology in Computer Science from Uttar Pradesh Technical University Lucknow. She is an active IEEE member. She has served as Trainer-Information Security in Mphasis, Bangalore as an intern. Her current research area includes cloud computing, IT Governance, Risk & Compliance.

Venkateshwaran K received his B.Tech in Information and Technology from Anna University, Coimbatore, Tamil Nadu, India. He is pursuing his Master of Science in Cyber Law and Information Security at Indian Institute of Information Technology, Allahabad. He worked as Software Engineer for two years in the domain of Mainframe Systems. His current research area includes cloud computing, Data Protection and Privacy, Risk assessment.

Utkarsha Dikshit is a MS (Cyber Law and Information Security) research student in Indian Institute of Information Technology, Allahabad. She has completed her Bachelor of Technology from SRM University, Chennai.