2010 Second International Conference on Computer Modeling and Simulation

Trust Level and Routing Selection for Mobile Agents in a Smart Home

Sandy Nasution[1], Pieter Hartel[2], Nanna Suryana[3], Nur Azman[4], Shahrin Shahib[5]
Faculty of Information and Communication Technology
Universiti Teknikal Malaysia Melaka
Melaka, Malaysia
{[1]sandy_ra,[3]nsuryana,[4]nura,[5]shahrinsahib}@utem.edu.my, [2]pieter.hartel@utwente.nl

***Abstract*—the central security concern for systems where agents roam is how to establish trust in the agent. We present a Fuzzy Logic mechanism to calculate a level of trust and an optimal route for a mobile agent system in a smart home. The mechanism consists of two parts. The first part calculates a trust level at the platform side to decide which actions should be allowed to a visiting mobile agent. The second part calculates an optimal route at the mobile agent side to decide an alternative destination in the case of rejection by a platform. We provide examples from smart home scenarios, showing how flexible the proposed mechanism is. The simulation has been implemented using Matlab Simulink.**

Keywords-component; mobile agent; trust level; routing selection;fuzzy logic;smart home;security

I. INTRODUCTION

The strength of a mobile agent is that it can migrate from one host in a network to another host that contains an object with which the agent wants to interact. The agent can then take advantage of being co-located with the object. A well designed mobile agent system is fault tolerant, reduces network load, overcome network latency, allows agents to execute asynchronously and autonomously, adapts to the change of environment dynamically, and provides adequate security.

The security problem has two aspects: protecting the agent from malicious hosts and protecting the host from malicious agents. This paper focuses on the latter issue. We refer the reader to Alfalayleh et al [1] for a comprehensive survey of the former issue. More particularly, this paper addresses the problem of how to decide to what extent a host can trust a visiting agent, what to do when the host partially trust a visiting agent, and where the agent can go next when it is rejected by a host that does not trust the agent at all.

Our solution to the problem has a wide range of applications, but we find it convenient to focus the discussion on a concrete setting, for which we use the smart home environment. We will discuss our approach using the following scenario.

Consider a smart home scenario where the home owner, Dieva and her family, are out on vacation. Dieva's friend Sita is enrolled in the biometric front door lock, so that she can look into the house from time to time while Dieva and her family are away. The events that occur the first time that Sita goes to Dieva's home are:

1) Sita arrives at the front door (FD) which is noted by the "guest" agent, who will serve Sita while she is in the house. The initial Trust Level of the agent is Low, i.e. too low to override the biometric lock, but since Sita has the front door key, the door opens and she enters the house.

2) Sita is in, and the "guest" agent then travels to the Home Control System (HCS) to notify the HCS that there is a guest entering the house.

3) Two processes in step 3:

a) Sita is thirsty; she goes to the kitchen searching for water. The Kitchen door biometric lock does not recognize Sita since she is not enrolled for biometric authenticaion at the kitchen door (KD). Therefore, Sita is not allowed to enter the kitchen.

b) Since she was rejected at the kicthen door, Sita decides to watch TV for a while. The process is the same as at the kitchen lock, i.e the "guest" agent travels through the HCS to the TV, but this time the trust level of the agent is sufficiently high to excute a process that turns the TV on.

4) Three processes in step 4:

a) While watching TV, Sita tries to go to the kitchen for the second time searching for water. For the same reason as in step 3a, Sita's Trust Level is too low to override the kitchen biometric door.

b) Sita then tries to open the kitchen door through the home control system. However, she is also rejected there since it is necessary for her to log in before she is allowed to perform any tasks.

c) Sita continues to watch TV, and suddenly the smart phone (SP) rings. For the same reason as with the TV, Sita is allowed to answer the phone. It is Dieva and they talked briefly. Because of the voice recognition in the Smart Phone, the latter infers that Sitas is talking to Dieva, the home owner. Therefore, the Trust Level of the "guest" agent is raised to Medium.

5) After talking to Dieva, Sita is getting really thirsty and tries to get some water from the kitchen again. Thanks to the interaction with the smart phone, the agent is now endowed with a Medium level of trust and can start a process that overrides the biometrick lock on the kitchen door.

The illustration of the scenario above can be seen in Fig. 1 below. Dashed lines represent possible routes of the

978-0-7695-3941-6/10 $26.00 © 2010 IEEE
DOI 10.1109/ICCMS.2010.463

"guest" agent, and solid arrows represent actual routes travelled. The Internet (Int) will be used in scenarios to be presented later.

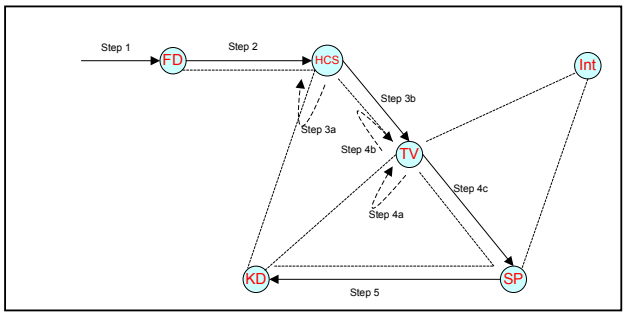

Figure 1. "guest" agent itinerary for the smart home scenario.

The scenario shows that on each occasion, the Trust Level of the "guest" agent is used by the platform as a basis for decision making. A visiting agent is only allowed to execute on a platform if its trust level is sufficiently high to meet the condition set by the platform. Similarly the agent must decide where to go, and what to do when rejected.

B. Problem

The problem is then twofold. The first aspect of the problem is how to calculate the trust level of an agent in such a way that neither the agent nor the platform can cheat. The second aspect is how to decide where the agent should go next in the case of rejection at a previous node, again without the agent or the platform being able to cheat.

C. Contribution

We propose to use fuzzy logic controllers to make the decisions, because this provides the necessary flexibility to accommodate all relevant decision factors [3]. To prevent agents and hosts from cheating, we show how existing methods from the literature can be used to protect the information on which the decision making is based. The overall approach is the following. A mobile agent that travels through many hosts will accumulate data as it travels. This data is presented to the next host to be visited and fed into a fuzzy logic controller to reach the decision. On the other hand, the agent has its own fuzzy logic controller that allows the agent to decide where to go next. This decision is based on information that an agent has about the platform it wants to visit. Both types of information that feed the fuzzy logic controllers must be appropriately protected.

Section 2 presents related work on calculating the trust level of an agent, followed by section 3, which discusses how to integrate the proposals from the literature with the fuzzy logic controller for the platform. Section 4 presents related work on agent routing, followed by section 5 which shows how the proposals from the literature can be integrated with the fuzzy logic controller for the agent decision mechanism. Section 6 evaluates our proposals by showing a number of simulation experiments from the home scenario. Section 7 shows how the simulation is implemented. Section 8 discusses a mechanism used to protect our decision factors, and the last section concludes and suggests future works.

II. Related Work on Calculating Trust Levels

The Primary purpose of agents is to gather information on their travels. When an agent platform decides what trust level to give to an agent, it must base its decision on appropriate factors. Here, we discuss a number of proposals for such factors, and we also suggest one of our own (Agent Behavior).

A. Path History

When an agent travels a multi-hop itinerary, it visits many platforms that are not all trusted to the same extent. The newly visited platform may benefit from the answer to the following questions: where has the agent been? How likely is it that the agent has been converted to a malicious agent during its trip? To enable the platform to answer these questions, a mobile agent should maintain an authenticated record of previously visited platform during its travel history. Using this history, the platform can make the decision whether to run the agent and what level of trust, services, resources and privileges should be granted to the agent [4, 5].

B. Time

Grimley and Monroe [6] use the factor of time to help identify a malevolent host. If the amount of time needed to execute a mobile agent on a host is limited, then the chance that it would become malicious is assumed to be minimized. Once the maximum amount of time needed by a mobile agent to execute safely on an entrusted host is exceeded, the agent must shut down or move to the next host specified on its itinerary. Therefore, the execution time is a useful decision factor.

C. Path Pattern

There are two types of mobile agent: a one-hop agent and a multi-hop agent. A one-hop agent is an agent that only migrates from one host to another and back, or it can also stay at the destination host. On the other hand, a multi-hop agent is a mobile agent that travels through many hosts [5]. For a multi-hop agent, the list of hosts visited is the path history.

Cao and Lu [7] represent the path history as a sequence of host identifiers like "$h_1h_2 \ldots h_n$". A path pattern is a regular expression that uses host identifiers as its alphabet to match a set of paths with some property. For example, "$*h_ah_b*$" matches the paths of the agents which have travelled to h_a and then directly to h_b.

D. Agent Behaviour

A more abstract decision factor is the agent behavior. An agent that is performing its task according to its normal routines is given a good mark. An agent behaving in a normal fashion always uses a correct path and reasonable amount of processing time to execute, while an agent is deemed to be bad if it behaves anomalously, for example when it makes a detour on the Internet for no good reason.

In our trust level calculations we use a number of the factors above by way of example, other factors can be accommodated too.

III. TRUST LEVEL CALCULATION

Based on the scenario from the introduction, and using some decision factors from section 2, we will show a number of examples of how an FLC can be used to make decisions on the trust level of an agent. The output of the FLC is a prediction of the level of trust attributed to the agent. We classify the level of trust as follows: High, Medium, and Low. This classification is then used by the host to determine the requirements put on the visiting agent, as well as the services, and resources granted to the visiting agent [2]. If the trust level of an agent is High, the agent is granted access to all resources of the host, the computing resources, memory usage and only simple authentication is required. If the trust level of an agent is Medium, the agent is allowed to execute but only granted a limited memory usage with a sufficient data access. However, the authentication process is sophisticated, and a security mechanism is being applied while the agent is executing i.e. Sandboxing, Proof carrying Code (PCC), or Code-Signing. The worst scenario is when the agent is deemed to be malicious as indicated by a Low trust level; in this case, the agent is rejected upon its arrival [2].

A. Refinement of the Scenario

Before we delve into the details of the FLC that supports the agent decision making, we discuss our running example again. Table I shows in more detail than Fig. 1 what the trust levels are at each of the 5 steps taken by the guest agent, and what the possible routes are at each stage.

TABLE I. DETAILS FOR THE SCENARIO

	FD	HCS	SP	TV	KD	Int	Stop
				Threshold			
Step	***10***	***90***	***50***	***50***	***60***	***80***	**-**
1	√	X	X	X	X	X	X
2	X	91.4>90 √	X	X	X	X	X
3 (a,b)	X	X	X	51.3>50 √(b)	51.3<60 X(a)	X	X
4 (a,b,c)	X	52.4<90 X(b)	52.4>50 √(c)	X	52.4<60 X(a)	X	X
5	X	X	X	X	63.2>60 √	X	X
6	X	X	X	52 > 50 √	X	X	X
7	X	X	X	X	79 > 60 √	X	X
8	X	X	X	64.2>50 √	X	X	X
9	X	X	X	X	X	X	√

Table I shows that the "guest" agent is triggered at the front door (step 1). From the front door the next (and only) destination is calculated, which is HCS. Upon arrival at the HCS, the trust level of the agent is 91.4, which is greater than the threshold 90, and thus high enough for the agent to be accepted at the HCS. A step 3, the agent selects KD as the first candidate to be visited (label a), but since the trust level of the agent, i.e. 51.3 is lower than the threshold 60, the agent is rejected and then migrates to the second candidate (label b), i.e. the TV where it is accepted. In step 4, the agent chooses KD as the first candidate (label a) and HCS as the second (label b) but gets rejected at the both sides, therefore the agent migrates to the third candidate (label c) SP where its trust level (51.3 > 50) is high enough to be accepted. Step 5, from SP the agent then tries to go to the KD again, and at this step, its trust level is sufficient to be accepted. The process continues until step 9 where the agent is terminated.

In this detailed scenario we can see that the agent is terminated at step 9 because its trust level (i.e. 9.7) is lower than the threshold of any of the devices inside the house. The trust level is getting lower each time the agent migrates. The factor that causes this is the length of the path taken by the agent (c.f Section 2.A). However, this is not realistic, because logically the longer a person (and the accompanying agent) stays in a house, the more trust should be granted to the person. Therefore, the path length is increased only when the agent steps out of the house (for example to roam the Internet). While staying inside the house, the path length is not increased. After changing the FLC to follow this more realistic path history rule, the simulation shows that the guest agent is never terminated, as long as it remains in the home. Space precludes us from showing the change to the FLC, but it is simple indeed, thus indicating that our approach using FLC is flexible.

B. The FLC

We present a high level description of the FLC that is used to calculate the Trust Level. The FLC that calculates the candidate destinations is discussed in section 5. The flexibility of an FLC makes it easy to add more rules and linguistic variables to embrace new decision factors.

In Section 2, we have discussed some of the decision factors that we believe to be useful in calculating the Trust Level.

Among all of those decision factors, we have taken Path length, Time, and Behavior as our parameters to be fed into the FLC.

The Path length is taken into consideration because it will show how far the agent has traveled, and we believe that the longer the path is, the higher possibility for the agent to be affected by any malicious entity.

We also take Time as one of our decision factor, because we believe that if an agent executes longer than normal, the possibility for that agent to be malicious is higher.

The Behavior decision factor tries to capture whether an agents behaves normally. For example, if an agent takes a detour, there is a possibility for that agent to be malicious.

These three decision factors are fed into the FLC to produce the trust level of the agent upon its arrival. After the trust level of this agent is calculated, the platform will update the previous decision factors based on the agent's activities.

The Path length will be updated by adding the number of hops traveled by the agent. The Time decision factor is updated based on the time taken by the agent to execute at the given host, and the behavior is updated by observing the path traveled, and the execution time of an agent at the given

host. For example, if the agent does not follow the path pattern or if it takes longer than normal, the agent is labeled with bad behavior. The Behavior decision factor can be updated based on any other decision factors, it is not only limited to Path length and Time.

Trust Level calculation requires a rule base and a set of equations. The elaboration of the equations and the rule base can be found in the extended technical report version of this paper [11].

IV. Decision Factors for Routing Selection

When an agent platform decides to reject a visiting mobile agent, it is critical for the agent to find an alternative platform it can visit. To decide on the alternative platform to be visited, the decision of the routing selection FLC must be based on appropriate factors. Here we discuss a number of proposals for such factors.

A. Number of Neighbors

In a smart home environment, the devices that are working together are connected via a wireless of wired network (indicated by dashed lines in Fig. 1). The mobile agents embedded in these devices are expected to migrate from node to node. Since the number of devices in a smart home is relatively small, it is reasonable to assume that each device knows the number of neighbors connected to it, and also the number of neighbors connected to its neighbors. We believe that the number of neighbors connected to the next visited nodes is important because in the case of rejection at the visited nodes, the mobile agent will have more alternative nodes to be visited. Marwaha et al [8] and Liu et al [9] use this decision factor in a wireless routing protocol. Here, we are trying to use it in a mobile agent environment.

B. Number of Accepts and Rejects

To decide the next destination of a multi-hop mobile agent, we believe that the number of Accepts and Rejects in the previous attempts is important. A mobile agent that travels through many platforms will keep records of each platform it has visited. These records also contain the information about the previous actions given to the mobile agent by the platform; it includes the reason for rejection or acceptance by the visited platform. In our protocol, the number of Rejects and Accepts at the previously visited nodes indicates the likelihood of the mobile agent to visit the same nodes again. If the number of the Accepts and Rejects is zero, the likelihood to visit this node is higher because it has not been visited before.

V. Routing Selection

Based on the scenario from the introduction, and the decision factors from Section 4, we show how an FLC can be used to determine the next destination of a mobile agent. The output of the FLC is a prediction of the best platform to be visited next. The FLC will take two major decision factors into account, which are: the number of Rejects and Accepts, and the number of immediate Neighbors connected to the next possible platform to be visited. Referring back to Table I, at each step one or more labeled boxes indicate the candidate destinations at this point. They are ordered by candidate number (a, b, c etc.), and they are tried in this order.

The number of candidates varies; it depends on the number of Neighbors of the platform where the agent is at present.

Now, let us consider the simulation from section 3, where the trust level of the agent is Low, which causes rejection at the KD when the agent tries to migrate for the first time from HCS to KD.

The rejection at KD (step 3a) will trigger the FLC at the agent side to present the next candidate destination to be visited. The FLC selects the TV (step 3b) as the second candidate. Therefore, the agent now tries to migrate to TV. Upon arrival at TV, the agent trust level is now calculated at TV side by using the trust level FLC and accepted at TV (step 3b).

A. The FLC

We present a high level description of the FLC that is used to calculate the Routing Selection.

Fig. 1 shows only a small collection of devices in a smart home. In a real smart home, there are more devices and more connections. Let us assume for example that KD is connected to 3 devices, i.e. TV, HCS, and SP. Therefore, the FLC now has to calculate which device (TV, HCS, SP) is the best candidate to be visited next.

Assume that TV has 3 neighbors and the agent has always been accepted at every attempt by the TV. HCS has 4 neighbors, and the agent has been rejected and accepted at least once in HCS. SP has 2 neighbors and it has been visited only once. Based on these decision factors the FLC will now decide which candidate neighbor is the most promising. The details of routing selection FLC can be found in the extended technical report of this paper [11].

After the best candidate is selected, the mobile agent then updates the information about all the candidates as the feedback for the future routing selection. The number of Neighbors decision factor does not change unless there is a new device is installed in the house. However, the number of Accepts and Rejects at each platform is updated because it will determine whether the mobile agent should select the platform again later.

VI. Other Scenarios

In this section we show three further scenarios of an agent serving a person in the smart home. Table II represents a guest agent that is roaming around the house and behaving oddly by migrating to the Internet (outside entity), where as Table III and IV represent an agent that acts for the home owner with various behaviors.

TABLE II. Guest Agent Travels by Migrating to the Internet (Outside Entity)

	FD	**HCS**	**SP**	**TV**	**KD**	**Int**	**Stop**
			Threshold				
Step	***10***	***90***	***50***	***50***	***60***	***20***	**-**
1	√	X	X	X	X	X	X
2	X	91.4>90 √	X	X	X	X	X

	FD	HCS	SP	TV	KD	Int	Stop
				Threshold			
3 (a, b)	X	X	X	51.3>50 √(b)	51.3<60 X(a)	X	X
4 (a,b,c)	X	52.4<90 X(b)	52.4>50 √I	X	52.4<60 X(a)	X	X
5	X	X	X	X	X	34.6>20 √	X
6 (a, b)	X	X	X	17.8<50 X(b)	17.8<60 X(a)	X	√

The first four steps in Table II are the same as in Table I. The differences begin when the "guest" agent migrates to the Internet which indicates unusual behavior of the agent (Step 5).

We have stated before that the path traveled by the agent inside the house can be considered as zero, while stepping out of the house should increase the path length. The Low trust level is calculated on the basis of this more realistic notion of path length causes the agent to be banned from any further interaction in the home.

TABLE III. HOME OWNER AGENT TRAVELS WITHOUT MIGRATING TO THE INTERNET (OUTSIDE ENTITY)

	FD	HCS	SP	TV	KD	Int	Stop
				Threshold			
Step	***10***	***90***	***50***	***50***	***60***	***80***	**-**
1	√	X	X	X	X	X	X
2	X	X	X	X	79.4>60 √	X	X
3	X	X	X	51.3>50 √	X	X	X
4	X	X	58.8>50 √	X	X	X	X
5	X	X	X	55.1>60 √	X	X	X
6	X	X	X	X	69.8>60 √	X	X
7	X	X	X	63.3>60 √	X	X	X
…	…	…	…	…	…	…	…

Table III above shows how the agent accompanying the home owner is accepted at every step by every device in the house. This special treatment is only applicable while the home owner agent is roaming inside the house.

TABLE IV. HOME OWNER AGENT TRAVELS BY MIGRATING TO THE INTERNET (OUTSIDE ENTITY)

	FD	HCS	SP	TV	KD	Int	Stop
				Threshold			
Step	***10***	***90***	***50***	***50***	***60***	***20***	**-**
1	√	X	X	X	X	X	X
…	…	…	…	…	…	…	…
8	X	X	56.3>50 √	X	X	X	X
9	X	X	X	X	X	56.3>20 √	X
10	X	X	71.3>50 √	X	X	X	X
11	X	X	X	57.2>50 √	X	X	X
…	…	…	…	…	…	…	…

Migrating to the Internet also reduces the trust level of the home owner. Most of the steps of Table IV are the same as Table III, and therefore not shown. The differences start at step 9 where the agent migrates from the phone (SP) to the Internet (Int) and back to the phone (SP). But since this agent represents the home owner, the agent still gets accepted at every device inside the house. The case of rejection only occurs when the agent extensively on the Internet, which causes its level of trust to be really low.

VII. IMPLEMENTATION

The implementation of the agent simulator has been written using Matlab 2008 Simulink. The steps of the simulation process are shown in Fig. 2.

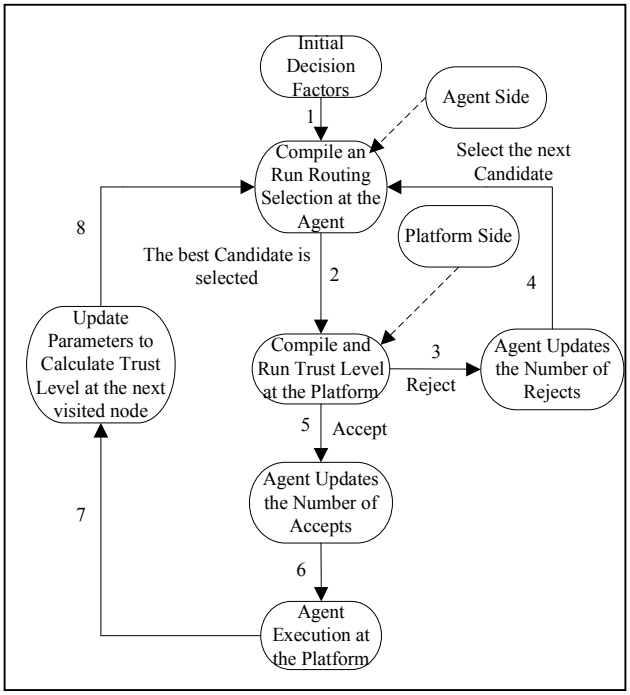

Figure 2. Simulation Process Flow Diagram.

The system contains two major subsystems, which are; the Trust Level FLC at the platform side, and the Routing Selection FLC at the mobile agent side. In the simulation we show how these subsystems are working together. The subsystem at the platform side decides whether to accept or reject a visiting agent, while in the other side, the Routing selection subsystem decides the next destination for a mobile agent in the case of rejection.

Fig. 2 shows the process flow of the simulation. At the start, the initial decision factors required for the trust level calculation are embedded in the agent (arrow 1). The agent then determines the next node to be visited by calculating the routing selection using the FLC at the agent side. After the next destination is determined, the agent then migrates to the destination while carrying the parameters (arrow 2). At the platform side, these parameters are used to calculate the trust level using the FLC.

The result from the trust level calculation is then used to determine whether the agent is accepted or rejected.

The agent that has been accepted (arrow 5) updates the number of Accepts of the visited platform and is allowed to execute at the platform. After executing, the platform then updates the decision factors of the agent for the next calculation (arrow 7).

In a case of rejection (arrow 3), the agent updates the number of Rejects of the respective platform and runs the routing selection again to choose the next best destination after the previous attempt (arrow 4).

The simulation goes on until the trust level of the agent is getting so low that the agent gets stuck, or until the user watching the simulation has been convinced that the two FLC's work as they are intended to.

VIII. Protecting Decision Factors

In this section, we discuss a mechanism to protect the decision factors that are used to calculate the trust level. Most mobile code systems consider the platform as a trusted entity and therefore they do not provide any mechanism to protect agent execution. In our system, the decision factors regarding the behavior of an agent are updated each time the agent finished executing. Therefore, we need a technique to protect the decision factors from being changed or tampered by any party.

If such tampering occurs, we would like to know any possible unauthorized modification of the agent code or state. However, we cannot predict the information that an agent will receive from the platform it is running on or from other agents.

As the solution for this problem, we suggest to use the Cryptographic traces method introduced by Vigna [10]. The proposed protocols assumes that all the involved principals, namely users and site owner or in our case platform and the home owner, own a public key and a secret key that can be used for encryption and digital signatures. Further explanation about this mechanism can be found in [10].

IX. Discussion and Future Work

In previous work [2] we have provided an overview of mechanisms designed to protect mobile agent platform from malicious mobile agents by calculating the mobile agent trust level. The trust level is used to decide actions that should be given to the mobile agent when it arrives at a platform. These actions are categorized into three categories. One of these actions is to reject the incoming mobile agent. Then a question of how the mobile agent should react in the case of rejection is raised, and what alternatives actions should it performs for a given situation.

We argue that the mobile agent should be able to migrate to another platform in the case of rejection at the previous platform. We have used a Fuzzy Logic Controller as a defense mechanism for mobile agent routing selection. In this paper, the simulation of how these two decision mechanisms are interrelated is also given.

Finally, we show by example of how Fuzzy Logic can be used with the relevant decision factors to perform routing selection for a mobile agent.

For future work, we would like to work out the examples in more detail, and build a full simulation of a smart home that supports a variety of realistic usage scenarios.

References

[1] M. Alfalayleh and L. Brankovic, "An Overview of Security Issues and Techniques in Mobile Agents," 8th IFIP TC-6 TC-11 Conference on Communications and Multimedia Security, pp. 59-78, Springer, 2004.

[2] S. Nasution, N. Suryana, S. Shahib, N.A Abu, and P.H Hartel, "The Application of Fuzzy Logic Controller to Compute a Trust Level for Mobile Agents in a Smart Home," In: International Technology, Education and Development Conference (INTED), 9-11 March 2009, Valencia, Spain. 702. International Association of Technology, Education and Development (IATED). ISBN: 978-84-612-7578-6. Available: http://eprints.eemcs.utwente.nl/15307/

[3] M. Ganesh, Introduction to Fuzzy Sets and Fuzzy Logic. 2006, New Delhi: Prentice-Hall of India.

[4] D. Chess, B. Grosof, C. Harrison, D. Levine, C. Parris and G. Tsudik, "Itinerant Agents for Mobile Computing," Technical Report, Oct. 1995, IBM T.J Watson Research Center, NY.

[5] J.J Ordille, "When Agents Roam, Who Can You Trust?," Proceedings of the Annual Conference on Emerging Technologies and Application in Communication, pp: 188-191, Portland, Oregon, May 1996, IEEE.

[6] M.J. Grimley and Monroe, "Protecting the integrity of agents: and exploration into letting agents loose in unpredictable world," Crossroads, 1999, 5(4), pp: 10 – 17.

[7] C. Cao, J. Lu, "A Path-History-Sensitive Access Control Model for Mobile Agent Environment," Proceedings of the Third International Workshop on Mobile Distributed Computing (MDC) (ICDCSW'05), vol. 6, pp. 660-663, June 2005.

[8] S. Marwaha, D. Srivasan, C.K. Tham, and A. Vasilakos, "Evolutionary Fuzzy Multi-Objective Routing for Wireless Mobile Ad Hoc Network," Congress on Evolutionary Computation, 2004, vol.2, pp. 1964-1971, 19-23 June 2004.

[9] H. Liu, J. Li, Y. Zhang, Y. Pan, "An Adaptive Genetic Fuzzy Multi-Path Routing Protocol for Wireless Ad Hoc Networks," In Sixth International Conference on Software Engineering Artificial Intelligence, Networking and Parallel/Distributed Computing, pp. 468-475, 2005.

[10] G. Vigna, "Cryptographic Traces for Mobile Agents," In Mobile Agents and Security, pp. 137-153, LNCS 1419, Springer-Verlag, June 1998.

[11] S. Nasution, P.H. Hartel, N. Suryana, N.A. Abu, and S. Shahib, "Trust Level and Routing Selection for Mobile Agents in a Smart Home (Extended version)" Technical Report TR-CTIT-09-36, Centre for Telematics and Information Technology, University of Twente, Enschede, 2009 Available: http://eprints.eemcs.utwente.nl/1600/