ISSN:0975-9646

Securing Mobile Agent Using Dummy and Monitoring Mobile Agents

Neelesh Kumar Panthi, Ilyas Khan, Vijay k. Chaudhari

Department of Information Technology, T.I.T. Bhopal,India
nee_panthi@yahoo.co.in
mikbpl_2003@yahoco.co.in
vijay_ashish@yahoo.com

Abstract
Mobile agent is a very important concept for distributed computing & utilizing the resources available on the connected network because of their capability of operation on different environments that is why the approach is used for many network based applications like data crawling, information exchange, distributed system integrity etc. but it lacks the security aspects when applied to open network where nodes cannot be classified as they are malicious or not, hence for the cases where the security of data or reliability of agent became must we need some method to insure the things discussed above. Although many other methods are proposed by many authors but some of them required a pre survey [1], encryption of important data [2] acknowledgement schemes [3], here we are proposing a scheme which not only confirms the security of data but also guarantees the uninterrupted operation of agent by utilizing a dummy agent and composite acknowledgement technique.

***Keywords*— Mobile agent, security, distributed computing.**

I. INTRODUCTION

Mobile agents are mobile autonomous processes operate on behalf of users in a distributed computing environment. The autonomous agent concept has been proposed for a variety of applications on large, heterogeneous, distributed systems (e.g., the Internet) [4]. These applications include a specialized search of a middleware services such as an active mail system, large free-text database [5], electronic malls for shopping, and updated networking devices. Mobile agent systems have many advantages over traditional distributed computing environments. They use less network bandwidth, increase asynchrony among clients and servers, dynamically update server interfaces and introduce concurrency [6].

Due to the problems with security of Mobile agents have limited their popularity. Mobile agents are composed of code, data, and state. Agents migrate from one host to another taking the code, data and state with them. The state information allows the agent to continue its execution from the point where it left in the previous host. For example, a mobile agent could be migrated from the home platform with the task of buying an airplane ticket for its owner. The agent would visit all the known hosts of airline companies, one after another, to search for the most reasonably priced ticket, and then purchase one for its owner. Each time the agent moves to the next host, it summarizes the current state, execution pointer on the current state, etc., so that it can start searching for reasonable tickets on the next host. The state of the agent will contain a set of possible tickets to be considered for purchase. When the agent has finished its search, it may return to the host where it found the cheapest or best ticket and purchase it.

While agents roam around the Internet, they are exposed to many threats and may also be a source of threat to others. Sander and Tschudin present two types of security problems that must be solved [7]. The first is host protection against malicious agents. The second is agent protection against malicious hosts. Many techniques have been developed for the first kind of problem, such as password protections, access control, and sand boxes, but the second problem seems to be difficult to solve. It is generally believed that the execution environment (host) has full control over executing programs; thus, protecting a mobile agent from malicious hosts is difficult to achieve unless some tamper-proof hardware is used. For example, Yee proposed an approach in which a secure coprocessor is used that executes critical computations and stores critical information in secure registers [8].

In this paper, we propose a security approach to protect mobile agents from malicious hosts. In this paper, to protect the agent data and agent itself we are proposing a method which not only protects the data but also agent. The proposed algorithm did not required previous travelling path of node and even the encryption is not required. Here we create agent with dummy data and service monitoring agent which generates the acknowledgment for host which is malicious or not.

The paper is structured as follows. In Section II we provide an overview of some previous works. In Section III we present our Algorithms starting with the underlying assumptions. Section IV shows the simulation results. In section V we briefly give conclusion for our approach.

II. PREVIOUS WORK

Mobile agent protection is difficult because of a host's complete control over executing programs. While many approaches have been proposed to defend mobile agents from malicious hosts, none adequately addresses every aspect of security. We survey three proposed approaches for the

problem of mobile agent protection. The three approaches are chosen because each approach is very uniquely implemented and has strengths that other approaches do not have; we choose Partial result authentication code approach because it can protect results from mobile agents. Computing with encrypted functions approaches is chosen because it tries to scramble code and data together. An obfuscated code approach is chosen because it scrambles an agent's code in such a way that no one is able to gain a complete understanding of its function.

A. Partial Result Authentication Codes (PRAC)

Yee [9] introduced Partial Result authentication Codes (PRACs). The idea is to protect the authenticity of an intermediate agent state or partial result that results from running on a server. PRACs can be generated using symmetric cryptographic algorithms. The numbers of encryption keys are used by agent. The agent's state or some other result is processed using one of the keys, producing a MAC (message authentication code) on the message when the agent migrates from a host. The key that has been used is then disposed of before the agent migrates. The PRAC can be verified at a later point to identify certain types of tampering. A similar functionality can be achieved using asymmetric cryptography by letting the host produce a signature on the information instead.

B. Computing with encrypted functions

This scheme is proposed by Sander and Tschudin [10] where an agent platform can execute a program embodying an enciphered function without being able to recognize the original function. For example, instead of equipping an agent with function *f*, the agent owner can give the agent a program P(E(*f*)) which implements E(*f*), an encrypted version of *f*. The agent can then execute P(E(*f*)) on *x*, yielding an encrypted version of *f*(*x*). With this approach an agent's execution would be kept secret from the executing host as would any information carried by the agent. For example the means to produce a digital signature could thereby be given to an agent without revealing the private key. However, a malicious platform could still use the agent to produce a signature on arbitrary data. Sander and Tschudin therefore suggest combining the method with undetachable signatures. Although the idea is straightforward, the trick is to find appropriate encryption schemes that can transform functions as intended.

C. *Obfuscated code*

Hohl proposes what he refers to as Blackbox security to scramble an agent's code [11] in such a way that no one is able to gain a complete understanding of its function. However, no general algorithm or approach exists for providing Blackbox security. A time-limited variant of Blackbox protection is proposed as a reasonable alternative. This could be applicable where an agent only needs to be protected for a short period. One serious drawback of this scheme is the difficulty of quantifying the protection time provided by the obfuscation algorithm.

III. PROPOSED ALGORITHM

Mobile agent is software written in platform independent language or package. Because of self mobility of software through which it can transfer to itself from one to another system connected in network. During this operation it can exchange the required data from each system as per requirement or according to script.

Because of mobility of agent it is very helpful for utilizing the network resources.

But the problem with this type of system is security of the agent, because it holds the important data and when it executes on some platform the platform takes on the complete control of it, and hence retrieves the complete data or can temper the agent

Hence to protect the agent data and agent itself we are proposing a method which not only protects the data but also agent. The proposed algorithm did not required previous travelling path of node and even the encryption is not required.

We can explain our algorithm in following steps-

Step 1: *The agent creates a duplicate agent with same script but with dummy dataset it also creates a services monitoring agent which can generate the acknowledgment.*

Step 2: *Agent transfers the dummy agent to next node with monitoring agent now the monitoring agent start monitoring the services of system if it defects that some critical services has started which can be used to temper the agent or data it generates an alert acknowledgment to main gent waiting in previous node.*

And if in case the monitoring agent gets tempered then it will not be able to send the acknowledgment in this case after some time the main agent decides to repeat the complete process again or escape the node.

Step 3: *If every thing goes ok then monitoring agent will send an ok acknowledgment to main agent which confirms the security of agent to same this node. The confirmation acknowledgment can be dynamically created to confirm the authentification of acknowledgment.*

A. Simulation Consideration

We have simulated the above discussed algorithm in OPNET modular 14 for calculation the overheads and execution efficiency, during the simulation we have taken the following considerations:

1. *Each agent is formed as packet so we have four types of packets.*

1. *main_agent*
2. *dummy_agent*
3. *Monitoring_Agent*

4. *ack _packet*

And the fields of the packets are shown in Figure 1.

2. *The services of system are a randomly assigned variable on which some numbers are considered as critical services.*

3. *The work of each agent is performed in node by some parts of their process model as the considered a part of agent.*

Main_Agent:

pkt_id	dest_id	send_id	server_id	ticket_id	card_id	exec_id	ip_list
8-bits	8-bits	8-bits	160-bits	160-bits	8-bits	8-bits	160-bits

Dummy_Agent:

pkt_id	dest_id	send_id	server_id	ticket_id	card_id	exec_id	ip_list
8-bits	8-bits	8-bits	160-bits	160-bits	8-bits	8-bits	160-bits

Monitoring_Agent:

pkt_id	dest_id	send_id	data_id
8-bits	8-bits	8-bits	8-bits

Ack_pkt:

pkt_id	dest_id	Send_id	data
8-bits	8-bits	8-bits	8-bits

Figure 1: Fields of Packets

4. *Tempering of data done on random basis by setting some flag in monitoring agent process model.*

5. *The topology of network considered as star.*

6. *For this simulation we did not limit the bandwidth of channel.*

IV. The Program Simulation Shows the Following Results

Result shown in Figure 2 shows the protection against malicious node (system protects the agent to travel through malicious node).

A. Simulation Result for Suspicious Nodes

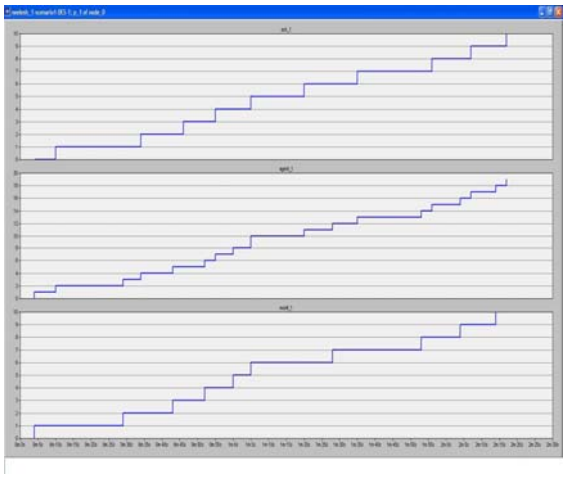

Figure 2: Simulation result for 12 nodes where 1 node is suspicious (total simulation time 300 seconds).

Conclusion is drawn on the basis that it sends total 10 monitoring & dummy agents and receives 10 acknowledgements but we have corrupted one node (to consider as it started some suspicious services) hence monitoring sends acknowledgement as suspicious node, when receiving node detects this type of acknowledgement it does not sends the original agent on this node and continues with next node hence number of original agents plus dummy agents reduces to 19 instead of 20.

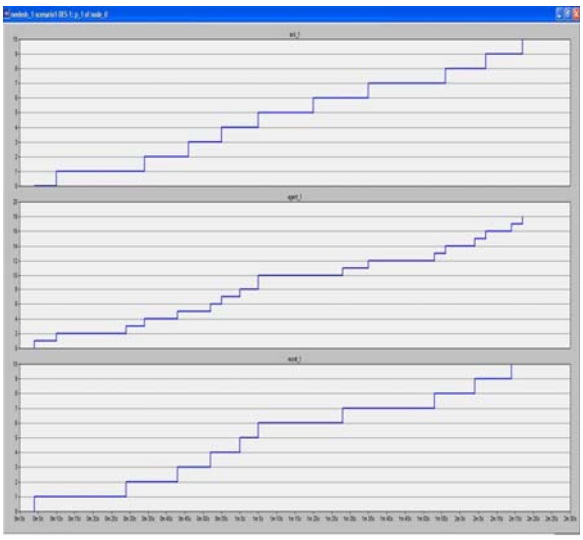

Figure 3: Simulation result for 12 nodes where 2 nodes are suspicious (total simulation time 300 seconds).

In Figure 3 the simulation shows 2 suspicious nodes are created and we get 18 agents count instead of 20 this verify the conclusion drawn according to first simulation results.

B. Simulation Result for Faulty Nodes

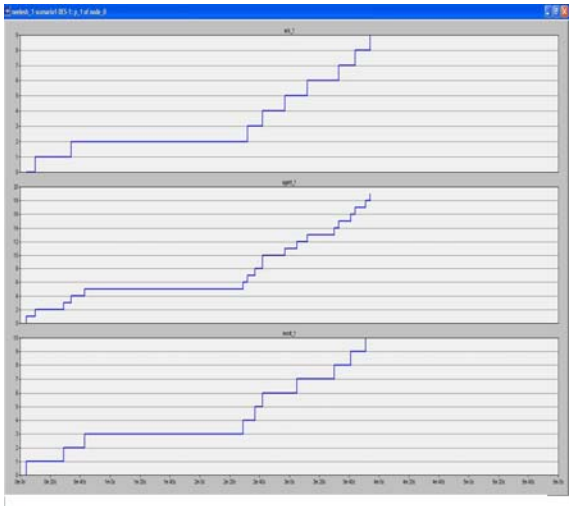

Figure 4: Simulation result for 12 nodes where 1 node is faulty (total simulation time 6.5 minutes).

Result shows the protection against faulty node (system protects the agent to collapse through faulty node or hanging in faulty node) shown in Figure 4.

Conclusion is drawn on the basis that it sends total 10 monitoring & dummy agents and receives 9 acknowledgements but we have corrupted one node (to consider as it grasp agent & monitoring agent) hence monitoring sends no acknowledgement, when receiving node did not detects acknowledgement for a certain time duration, it does not sends the original agent on this node & continues with next node hence number of original agents plus dummy agents reduces to 19 instead of 20.

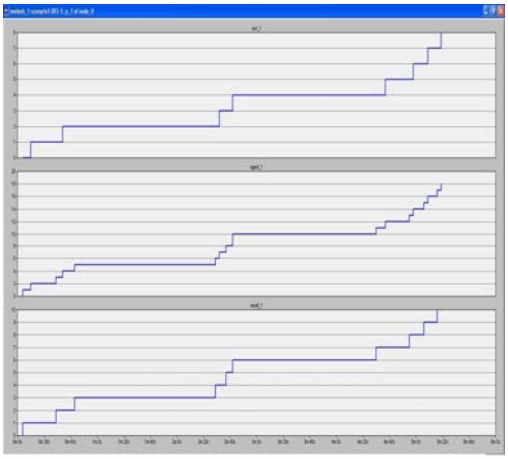

Figure 5: Simulation result for 12 nodes where 2 nodes are faulty (total simulation time 6.5 minutes).

In Figure 5 the simulation shows 2 faulty nodes are created and we get 18 agents count and 8 acknowledge count instead of 20 and 10 this verify the conclusion drawn according to first simulation results.

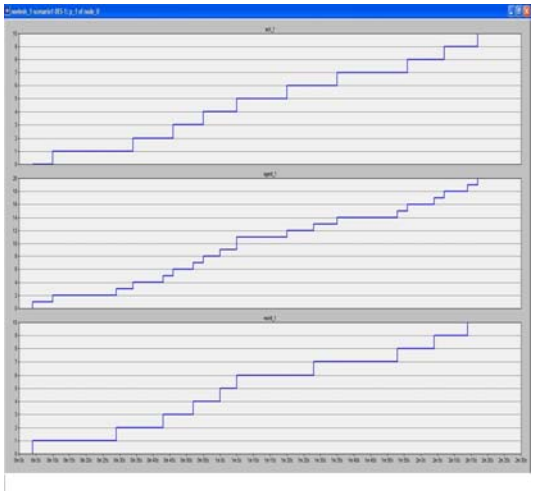

Figure 6: Simulation results for faultless conditions (total simulation time 2.2 minutes)

Result shown in Figure 6 shows the simulation for faultless network. There are all nodes are not faulty or suspicious.

Conclusion is drawn on the basis that it sends total 10 monitoring & dummy agents and receives 10 acknowledgements. We get 20 agents count out of 20 which shows all nodes are ok according to first simulation result.

V. CONCLUSIONS

The simulation model thus implemented is used to study the behaviour of mobile agents in some important Internet applications where they are believed to have better performance. The analysis of the results thus obtained shows the protection of mobile agent against malicious host and very high chances of successful completion of task and depends only on bandwidth of system and time out limit of agent. We can further enhance it by applying retry on time out. The simulation also shows the no agent blocking for any number of faulty nodes.

Some draw back shows the increase in delay, but this can be overcome by flooding multiple dummy agents to multiple nodes simultaneously. The overhead and process utilization can be minimized by reducing the size of dummy agent.

REFERENCES

[1] Ichiro Satoh. *Selection of Mobile Agents*. In Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04). IEEE Computer Society Press, 2004.

[2] J. White, "*Mobile Agents White Paper*," General Magic Inc., 1996.

[3] D. Milojici, "*Mobile agent applications*", IEEE concurrency, July-Sep 1999, pp 80- 90.

[4] Chandra Krintz, *Security in agent-based computing environments using existing tools*. Technical report, University of California, San Diego, 1998.

[5] Joshua D. Guttman and Vipin Swarup. Authentication for mobile agents. In LNCS, pages114–136. Springer, 1998.

[6] Neeran Karnik. Security in Mobile Agent Systems. PhDthesis, Department of Computer Science and Engineering. University of Minnesota,1998.

[7] Tomas Sander and Christian F. Tschudin. Protecting Mobile Agents Against Malicious Hosts.In Giovanni Vigna, editor, Mobile Agent Security, pages 44–60. Springer-Verlag: Heidelberg,Germany, 1998.

[8] Bennet Yee. Using Secure Coprocessors. PhD thesis, Carnegie Mellon University, 1994.

[9] Bennet Yee. A sanctuary for mobile agents. In J. Vitek and C. Jensen, editors, Secure Internet Programming, volume 1603 in LNCS, pages 261–274, New York, NY, USA, 1999. Springer-Verlag Inc.

[10] Tomas Sander and Christian Tschudin. Towards mobile cryptography. In Proceedings of the IEEE Symposium on Security and Privacy, pages 215–224, Oakland, CA, May 1998. IEEE Computer Society Press.

[11] Tomas Sander and Christian F. Tschudin. Protecting Mobile Agents Against Malicious Hosts.In Giovanni Vigna, editor, Mobile Agent Security, pages 44–60. Springer-Verlag: Heidelberg,Germany, 1998.

[12] Fritz Hohl. Time limited blackbox security: Protecting mobile agents from malicious hosts. In G. Vigna, editor, Mobile Agents and Security, volume 1419 in LNCS, pages 92–113. Springer-Verlag, Berlin, 1998.